# Distributed Monitoring For Intrusion Detection In Clouds


**Sultan Saad Alshamrani**

Department of Computer Science

University of Liverpool

Thesis submitted in accordance with the requirements of
the University of Liverpool for the degree of Doctor in Philosophy.

*Doctor of Philosophy*

January 2017

To my parents.

اهداء الي والدي

# Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this PhD thesis is original and have not been submitted in whole or in part for consideration for any other degree or qualification in the University of Liverpool, or any other university. This thesis is the result of my work and includes nothing which is the outcome of work done in collaboration, except where explicitly indicated.

The main part of this thesis is based on seven papers, which have been peer reviewed and accepted to different research conferences. One of these papers is submitted to the special issue in Recent Patents on Computer Science Journal. The full list of papers in which I contributed on equally proportionate level.

1. Sultan S. Alshamrani, Dariusz R. Kowalski and Leszek A. Gasieniec, "The Impact of Hierarchical Structure on Efficiency of Cloud Monitoring ", submitted to a special issue of Recent Patents on Computer Science Journal. This paper is a selected paper that originally presented in Cloudtech 2016. [1].

2. Sultan S. Alshamrani. "Discovering Malicious Behaviour Symptoms in Cloud Systems". In: Proceedings of the Eighth Saudi Students Conference in the UK. 2016. p. 375-384. [2].

3. Sultan S. Alshamrani, Dariusz R. Kowalski and Leszek A. Gasieniec, "Efficient Discovery of Malicious Symptoms in Clouds via Monitoring Virtual Machines", 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), vol., no., pp.1703-1710, 26-28 Oct. 2015 doi: 10.1109/CIT/IUCC/DASC/PICOM.2015.257. [3].

4. Sultan S. Alshamrani, Dariusz R. Kowalski, Leszek A. Gasieniec and Muhammed B. Abdulazeez, "Balancing mobility algorithm for monitoring virtual machines in clouds", in ECCWS2016-Proceedings fo the 15th European Conference on Cyber Warfare and Security, p. 9-18. [4].

5. Sultan S. Alshamrani, Dariusz R. Kowalski and Leszek A. Gasieniec, "The impact of hierarchical structure on efficiency of Cloud monitoring". In IEEE 2016 2nd International Conference on Cloud Computing Technologies and Applications (CloudTech) (pp. 40-46). [5].

6. Sultan S. Alshamrani. "How Reduce Max Algorithm Behaves with Symptoms Appearance on Virtual Machines in Clouds". In: 2015 IEEE International Conference on Cloud Computing (ICCC). 2015. p. 1-4. [6].

7. Sultan S. Alshamrani, Dariusz R. Kowalski and Leszek A. Gasieniec, "Random-Start-Round-Robin Algorithm for Monitoring VMs in Clouds", the paper is ready for submission [7].

Sultan Saad Alshamrani
January 2017

# Acknowledgements

This thesis is the end of almost three and half years journey to obtain my PhD degree in Computer Science. First and foremost, all praise be to Almighty God, Allah, the Most Gracious and Most Merciful, for giving me the power, the strength and the patience to overcome all challenges that came my way. He is also blessing me with those wonderful people who have believed in my abilities and have believed in me that I will pass my PhD journey easily and fruitfully. The inspiration for me to become an educational man came from Allah Almighty by the first word "Read" that came down from him to his messenger Prophet Mohammad in Holly Quran.

I am deeply indebted to my supervisor, Professor Dariusz Kowalski, who gave me the chance to work under his supervision and who has supported me with his invaluable assistance and guidance throughout my PhD research. He has been a great supporter and the researcher that listens, discusses and suggests research ideas. Through his extraordinary experience, he has taught me not only to be a good student but also a good researcher and an intellectual person. He has enlightened my way by his inspiration and endless efforts on how to explain and present academic work simply and clearly. He has always been a real friend. He was the perfect resource, which inspired me and enriched my experience. It has been a pleasure to have been supervised by him.

My deepest thanks and appreciation goes to my second supervisor, Professor Leszek Gasieniec, for his assistance, constructive suggestions, feedback, insightful comments and research ideas. He also supported me especially with ideas that helped me to do research.

Besides my supervisors, I would also like to extend my thanks to my advisors Professor Prudence Wong and Dr Russell Martin for their evaluation, constructive comments and feedback. I also want to thank my fellow PhDs who stood by my side, provided me with incredible support and had always been there for me when I needed them. In particular, Dr Mohamed Arikiez, David Hamilton, Muhammed Bello Abdulazeez and Abdullah A. Alshehri.

It is very difficult for me to find the right words to express my gratitude and thanks to my parents. They not only have raised me with endless care, love and support, but also they have constantly encouraged me to be an independent and self-confident person. Without their support and encouragement, this work would not have been started.

I am eternally indebted to my wife for her support and encouragement, respect, love and appreciation, without whom it would not have been possible to accomplish of my PhD. She has always been my best friend.

I shall not forget to thank my sponsors, University of Taif and the Royal Embassy of Saudi Arabia with the Cultural Bureau in London, for providing me with the opportunity to conduct my PhD study.

# Abstract

This thesis is in the field of Computer Science. More precisely, its main research themes are in the applied part of the field Cloud Computing. The main focus in this work is on monitoring of cloud systems in a distributed fashion.

This work is a natural continuation of previous studies on discovering the symptoms malicious behaviours in cloud systems. Our line of research is based on efficient discovery of the symptoms of threats. This challenge is met through the design and analysis of new algorithms carrying out this job.

Several algorithms are studied. First, a simplified version of previously studied Mobility algorithm is proposed. The new algorithm is named Reduce-Max algorithm. This algorithm is analysed on eight different data sets. Then two modifications of Reduce-Max algorithm are considered. The first one is called Randomised-Local Reduction and the second one is Deterministic-Centralised Reduction. Further, the algorithms are tested under different models of symptoms appearance. The work continues with studies of Reduce-Max and its two modifications in hierarchical systems, which concludes in the design of a new algorithm, called Random-Start-Round-Robin. Finally, this thesis concludes with work on balancing Mobility Algorithm.

An integral part of my PhD work are experiments of proposed algorithms where the emphasis is on proper modeling of monitoring of cloud systems. Further discussion is based on the results of these experiments reflected in the final conclusions.

# Contents

# List of Figures

# List of Tables

# Nomenclature

**Roman Symbols**

| | |
|---|---|
| $\alpha$ | Given value from the range (1, 1.2, 1.4, 1.6, 1.8, 2, 2.2, 2.4, 2.6, 2.8, 3) in each time of simulation |
| $\lambda$ | Vector of weights |
| $\mu$ | A given parameter that poisson distribution is based on |
| $c_i$ | Configuration $i$ |
| $Disc(c_i, v)$ | Number of symptoms in one configuration $c_i$ discovered at VM $v$ |
| $DOS$ | Denial of Service |
| $k$ | Number of configurations |
| $M$ | Number of FVMs |
| $N$ | Number of VMs |
| $OS$ | Operating System |
| $r$ | Round $r$ in the execution of the simulation |
| $size(c_i)$ | Number of potential symptoms in a configuration $i$ |
| $T$ | Time from last visit to VM $v$ (latency) |
| $v$ | A specific Virtual Machine $v$ |
| $val(c_i)$ | Assigned value of a configuration $i$ |
| FVM | Forensic Virtual Machine |
| VM | Virtual Machine |

# Part I

# Preliminaries and Foundations

# Chapter 1

# Introduction

## 1.1 Overview

The work in this thesis, as described in the abstract, is in the area of Distributed Systems, particularly in Cloud Computing Systems and more generally in the field of Computer Science. More precisely, the attempt is to study cloud monitoring problems with the goal of discovering malicious behaviours in cloud systems. Namely, the research approach focuses on determining malicious behaviour via discovering relevant symptoms rather than by targeting more specific malicious behaviour directly.

Cloud computing systems are often seen as dynamic pools of Virtual Machines (VMs) installed on a provider side physical machine and to be offered to cloud users [5]. In [8], cloud computing is defined as a model for enabling ubiquitous, convenient, on-demand network access to a shared environment of configurable computing resources.

The access to cloud computing systems is often granted to numerous users of Virtual Machines and to services via the Internet [9], [10]. This form of access makes cloud systems more vulnerable with respect to security than physical networks [4]. This is often a reason for more frequent and more serious attacks. In order to prevent or minimise the extent of attacks, and in turn to secure data storage, any malicious behaviour such as external undesirable interventions should be rapidly identified and halted if possible [2].

With the above goals in mind, this thesis is proposing and evaluating a number of algorithms targeting efficient symptoms monitoring in cloud systems. Such algorithms need to

identify the most appropriate ways to monitor Virtual Machines (VMs), for example via determining the best data sets, the most suitable number of Forensic Virtual Machines (FMVs) and the best selected ranges of highest valued VMs, all matched with the structure of cloud networks.

This introductory chapter is organised as follows: first, more detailed description of the motivation and what has been done in this thesis is presented in Section 1.2. This is followed by Section 1.3 which presents the main research question and related problems. The main research contributions and methods of evaluation of this thesis are listed in Section 1.4. The review of the published work to date resulting from the research described in this thesis is presented in Section 1.5. Section 1.6 refers to the organisation of the rest of this thesis.

## 1.2 Motivations

A strong motivation for this research comes from a number of security problems in cloud systems. The work covered by this thesis started through reading a number of seminal papers such as "A view of cloud computing" [11], "A survey on security issues in service delivery models of cloud computing" [12], "Addressing cloud computing security issues" [13], "On technical security issues in cloud computing" [14] and "On Cloud computing: issues and challenges" [15]. On the basis of these papers, it has been decided to focus on studies on symptoms like modifying the time attributes [16], service mix attributes [17], and identifying suspicious snippets of code [18] which indicate threats to cloud systems.

In particular, the paper titled "A framework for detecting malware in cloud by identifying symptoms" [18] lays down a good direction for research explorations suggesting Mobility algorithm as the most relevant tool for symptoms discovery.

These previous papers lead to the conclusions that the topic of cloud security issues has an increasing number of researches who work on it. The reason is that the work on cloud systems needs continuous developments on security due to the on-line access to those systems.

Finally, another reason for choosing this topic is its relation to symptoms discovery in human diseases. For example, when patients come to hospitals, the doctors first check basic symptoms, which help them to arrive in diagnosis of a specific disease or to determine possible causes of illnesses [18].

## 1.3 Research question and related issues

As described in this introductory chapter, it is very important to secure the use of cloud based data storage. The access to the data storage in clouds is mainly available via the Internet. Because the Internet is open for everyone, the threats of malicious behaviour become higher.

The aim of this research is to design and evaluate new algorithms for distributed decentralised monitoring of cloud systems. These algorithms target efficient discovery of symptoms of malicious behaviours in cloud systems.

The problem considered in this thesis is non-trivial due to the large scale of cloud computing systems in which often only small resources are available to support security and monitoring mechanism. Another important issue is the unpredictability of malicious behaviour in dynamic cloud systems.

The main question can be rephrased as what are the most suitable distributed algorithms for monitoring VMs with the goal of detecting intrusions.

## 1.4 Research contributions and evaluation processes

The main contributions of this thesis refer to the design and evaluation of new algorithms that support cloud monitoring by discovering symptoms of malicious behaviours. The work presented in this thesis was completed under supervision of my coauthors and research advisors. After starting with an introductory part of two chapters, a number of algorithms are proposed in Chapter 3. The evaluation environments for each algorithm are analysed in Chapter 4, and the outcomes are discussed in Chapter 5. This is followed by comparisons and evaluations of those algorithms in Chapter 6 and conclusions of the whole work in Chapter 7.

A thorough evaluation of research explorations is an integral part of any research activity. There are different ways to perform evaluation. First one is to compare the own work with other researchers' works in the same area of research [19]. Other methods include statistical testing via experiments with an adaptive interim analysis [20] and qualitative research evaluations [21]. In this thesis, the second and third approaches listed above will be adopted. However, for first approach a comparison between the thesis's algorithms themselves and the recent paper [22] is one of promising direction of this research.

## 1.5 Published work

In this section, we list publications relevant to the main theme of this thesis:

### 1.5.1 Journal paper

- Sultan S. Alshamrani, Dariusz R. Kowalski and Leszek A. Gasieniec, "The Impact of Hierarchical Structure on Efficiency of Cloud Monitoring", submitted to a special issue to Recent Patents on Computer Science Journal. This paper is a selected paper that originally presented in Cloudtech 2016 [1].

  Cloud computing systems are often seen as dynamic pools of Virtual Machines (VM) installed on provider side physical machines to be offered to Cloud users. Cloud customers could use these Virtual Machines as services, platforms or as a whole infrastructure. However, in practice the infrastructure of a computing cloud includes several levels, such as virtual gateways, virtual clusters and virtual nodes. In this paper, we pursue a study of the impact of a hierarchical structure, formed of three levels, on the process of monitoring the system with the main goal of discovering symptoms of malicious behaviours in clouds.

  We address in this paper two major questions. The first question refers to the optimisation of the number of clusters in the hierarchical structure to guarantee efficient monitoring.

  The second question, posed in some previous papers in this area, concerns efficient distributed implementation of the monitoring process; namely, how to choose locally the next VM to be visited by a Forensic Virtual Machines (FVM) in a light and local way. Note that this paper is for a special issue in the journal. This paper was originally presented in Cloudtech 2016.

### 1.5.2 Conference papers

The following conference papers have been published:

- Sultan S. Alshamrani. Discovering Malicious Behaviour Symptoms in Cloud Systems. In: Proceedings of the Eighth Saudi Students Conference in the UK. 2016. p. 375-384. [2]

Cloud computing system defined as a modern technology that enables users to share resources in a virtual storage and computing environment. Moreover, cloud system provides an environment of a number of virtual machines (VMs), which are used by multiple users, and implemented on a single physical server. In this paper, the approach focuses on identifying malicious behaviour via discovering respective symptoms rather than targeting particular malicious behaviour directly. The reason behind our research approach is that a malicious behaviour can be characterised by a set of symptoms. Finally, this paper provides experiments of monitoring VMs in order to discover symptoms.

In this paper, a new algorithm that is called Reduce-Max algorithm is proposed for the monitoring purposes and to overcome limitations in monitoring in previous work in [18]. Five configurations of data sets were considered, implemented and analysed for This Reduce-Max algorithm. They are (uniformly) random, uniform, arithmetic, harmonic, and exponential.

- Sultan S. Alshamrani, Dariusz R. Kowalski and Leszek A. Gasieniec, "Efficient Discovery of Malicious Symptoms in Clouds via Monitoring Virtual Machines", 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), pp.1703-1710, 26-28 Oct. 2015 [3].

  The main contribution of this paper refers to several new mechanisms for monitoring Virtual Machines and further experimental work targeting efficient ways of visiting VMs in order to discover malicious symptoms.

  In [3], there was a further development of Reduce-Max algorithm supported by a deeper analysis. The paper distinguished two ways of implementing Reduce-Max algorithm in case of many FVMs: deterministic coordinated reduction and randomised local reduction. The same five configurations of weights as in [2] were tested for Reduce-Max algorithm in both the considered versions (deterministic and randomised).

- Sultan S. Alshamrani, Dariusz R. Kowalski, Leszek A. Gasieniec and Muhammed B. Abdulazeez, "Balancing mobility algorithm for monitoring virtual machines in clouds", in ECCWS2016-Proceedings fo the 15th European Conference on Cyber Warfare and Security, p. 9-17. 2016. [4].

  In this paper, the focus is on the discovery of malicious behaviour via determining unwanted symptoms rather than via targeting particular malicious behaviours of the sys-

tem directly. The main contribution of this paper consists of several new mechanisms for monitoring Virtual Machines and further experimental work targeting efficient ways of visiting VMs in order to discover malicious symptoms. The goal is to find the fastest and the best set of weights for visiting VMs.

The main part focuses on the explanation of the work conducted by the authors on monitoring Virtual Machines. The balance of Mobility algorithm will be by setting two weights in front of each part of the mobility formula. These two weights have a sum of one.

- Sultan S. Alshamrani, Dariusz R. Kowalski and Leszek A. Gasieniec, "The impact of hierarchical structure on efficiency of Cloud monitoring". IEEE 2016 2nd International Conference on Cloud Computing Technologies and Applications (CloudTech) (pp. 40-46).2016. [5].

  In this paper, cloud customers could use Virtual Machines as services, platforms or as a whole infrastructure. However, in practice the infrastructure of a computing Cloud includes several levels, such as virtual gateways, virtual clusters and virtual nodes.

  The purpose is to study the impact of a hierarchical structure, formed of three levels, on the process of monitoring the system with the main goal of discovering symptoms of malicious behaviours in Clouds. We address in this paper two major questions. The first question refers to optimise the number of clusters in the hierarchical structure to guarantee efficient monitoring. The second question, posed in some previous papers in this area, concerns efficient distributed implementation of the monitoring process; namely, how to choose locally the next VM to be visited by a Forensic Virtual Machine in a light and local way.

- Sultan S. Alshamrani. How Reduce Max Algorithm Behaves with Symptoms Appearance on Virtual Machines in Clouds. In: 2015 International Conference on Cloud Computing (ICCC). IEEE, 2015. p. 1-4. [6].

  There is a continuation of the research that focuses on the discovery of malicious behaviour symptoms. The main contribution of this paper is the simulation analysis of algorithm Reduce Max with respect to its delay in monitoring cloud systems with different demand functions according to the appearance of symptoms that should be on a random occurrence.

  This attempt was made to examine that Reduce-Max algorithm is behaving well with the appearance of symptoms. This leads to early detection of malicious behaviour of

cybercriminals in clouds. The results of that work showed additional advantages of Reduce-Max algorithm.

- Sultan S. Alshamrani, Dariusz R. Kowalski and Leszek A. Gasieniec, "Random-Start-Round-Robin Algorithm for Monitoring VMs in Clouds", the paper is ready for submission [7].

  Cloud computing systems are pools of Virtual Machines (VM) installed on provider side physical machines to be offered to systems' users. To keep these systems secured, they should be monitored properly. To do this job, a number of algorithms were proposed previously. The work in this paper follows the work of previous algorithms. Therefore, we introduce a new algorithm that proposed to be helpful for the job of visiting VMs in regards to detect threats.

The work described in this thesis has also led to a number of other investigations, not reported in this thesis, which in turn has resulted in further publications (to which the author has made some contribution). For, completeness these publications are summarised as follows:

- M. B. Abdulazeez, D. R. Kowalski, A. Lisista, S. S. Alshamrani. "Failure or Denial of Service? A Rethink of the Cloud Recovery Model". in ECCWS2016-Proceedings of the 15th European Conference on Cyber Warfare and Security, p. 1-8. 2015. [23]

  One of the major paradigms of cloud computing is infrastructure as a service (IaaS), which allows organisations to outsource computing equipment and resources such as servers, storage, networking, as well as services such as load balancing and content delivery networks. A critical aspect and selling point of the vendors offering IaaS is load balancing. One component of load balancing is auto-scaling. This feature allows applications to scale up and down dynamically based on load, performance and 'health' of a virtual machine (VM). It used to take years to grow businesses to millions of customers but now this can happen in months or even days, therefore the ability of have infinite amount of resources on demand is a very appealing one to businesses. The entire cloud model relies on dynamic scalability and configurability, because it is not practical to manually configure such services. In this paper we propose a rethink of the way a cloud scales up, as vendors do not formally define what healthy means. We also look at application layer denial of service (DOS) attacks on application servers running compute services. While there has been extensive efforts to defend the cloud against volumetric DOS using network layer defenses. Detecting and preventing application layer DOS attacks on cloud however, is not trivial due to the size of cloud and the heterogeneity of

applications running. We surveyed some of the key cloud providers that offer IaaS such as Amazon Web Services, Windows Azure, Google Compute Engine, Rack Space Open Cloud and IBM Smart Cloud Enterprise. We specifically analysed their auto-scaling features and looked at the cost implications to customers. We ask the question, does the monitoring feature of these services differentiate between load increase and Application Layer DOS when making decision to scale up its services VM.

## 1.6 Thesis organisation

In this section, there is a presentation of the overall organisation of the thesis beyond this introductory chapter:

**Chapter 2** elaborates further on the outline provided in this chapter including further details. It provides the necessary background to the work described, together with a review of the related work. Among the topics covered in this background Chapter 3 are algorithms and distributed systems, cloud computing (including cloud security, cloud monitoring and cloud structures), symptom discovery and finally some of the previous work on mobility algorithms.

**Chapter 3**, located in the second part of this thesis, discusses in greater detail each algorithm starting with Reduce-Max algorithm and its two variants: Deterministic-Centralised and Randomised-Local versions. Moreover, this chapter provides further explanations to algorithms like Random-Start-Round-Robin algorithm. This chapter includes also description of the hierarchical structure for cloud systems monitoring. It include also a section about symptoms appearance with Reduce-Max, and finally details of balancing mobility algorithm.

**Chapter 4** is also in the Algorithms and Methodology Part. It provides evaluation environment of each algorithm that is presented in Chapter 3. In addition, it includes all data sets, called configurations, that are used for each explained algorithm.

**Chapter 5** is the first chapter of the final part of this thesis. This chapter contains all experimental results for each algorithm. These results are discussed in details in this chapter.

**Chapter 6** contains discussions, evaluations, and comparisons among results of algorithms in Chapter 5 to have a clear conclusions.

**Chapter 7** shows a summary of the main and final conclusions of these works and suggestion of potential directions for future work.

**Appendix A** includes tables that showing all experiments results in numbers which presented in figures in Chapter 5.

**Appendix B** includes all awards and some certificates obtained due to the work of this thesis.

# Chapter 2

# Background Work and Literature Review

## 2.1 Overview

This chapter presents a review of the background research and works relevant to the results presented later in this thesis. As explained earlier, the aim of this thesis is to design and evaluate new algorithms for performing distributed monitoring of cloud systems in order to discover malicious behaviours symptoms. This chapter provides reviews about each concept that is mentioned in the aim of this thesis.

The chapter is organised as follows: Section 2.2 describes a background research about algorithms. The reason for starting with algorithm section is that this thesis as mentioned in Chapter 1 proposes new algorithms. This section of algorithms is followed by a background research of distributed systems as cloud computing system is one of the examples of these systems. Another reason is that our algorithms are monitoring the cloud systems distributively. Section 2.4 discusses cloud computing concept which is divided into three subsections as follows: general knowledge about cloud computing, some literature in cloud security and mainly subsection about cloud monitoring. The last two sections in this chapter provide literature review in symptoms discovery area and the reviews of some papers such as [18] that proposed the mobility algorithm.

## 2.2 Algorithms

Historically, the word “algorithm” is a descendant of AlKhwarizmi from Abu Abdallah Muham-

Figure 2.1: A page from Al-Khwarizmi book “Al-kitab al-mukhtasar fi hisab al-gabr wa lmuqabala”

mad ibn Musa AlKhwarizmi (Father of Abdullah, Mohammad, son of Moses, native of Khwarizm), geographer and mathematician [24]. One of Al-Khwarizmi’s most famous works is the “Al-kitab al-mukhtasar fi hisab al-gabr wa lmuqabala” (“The Compendious Book on Calculation by Completion and Balancing”), which is also where the word “Algebra” (Al-gabr) originates from [25]. Figure 2.1 contains a page from this book. Now let us specify what an algorithm is.

**Definition 2.1.** [26] An algorithm is a procedure or set of rules used in calculation and problem solving; a set of mathematical or logical operations for the performance of a particular task.

From this definition and also from other sources such as [27], algorithms are ways of solving problems using step-by-step procedures. In [28], algorithms are used to solve real-world problems. In this way of defining algorithms, like the work in this thesis, several research papers propose new algorithms to solve various real-world problems. The readers can find some examples in [29–33]. All these works describe and propose algorithms for different research problems.

Finally, the work in this thesis is inspired by Mobility algorithm which is a heuristic described in details in Chapter 3. This work itself proposes the following algorithms; Reduce-Max algorithm, Deterministic-Centralised version of Reduce-Max algorithm, Randomised-Local version of Reduce-Max algorithm, Random-Start-Round-Robin algorithm and some models that are related to algorithms such as symptoms-appearance for Reduce-Max, the hierarchical structure model, and finally balancing mobility algorithm itself.

## 2.3 Distributed systems

In this section, some historical and background work about distributed systems are provided. The relevance of this topic is motivated by the fact that the work of this thesis is about distributed monitoring of virtual machines, and that cloud computing system itself is an example of distributed systems. Next paragraphs describe what distributed systems are in order to understand clouds better.

**Definition 2.2.** [34] A distributed system is a model in which components located on networked computers communicate and coordinate their actions by passing messages.

Another definition of distributed system is that it is a collection of independent computers that appear to its users as a single coherent system [35–37]. To explain this definition, we could see in Figure 2.2, distributed systems have a shared middle-ware, whatever it is, among different networked computers with different Operating Systems (OS), although they constitute one system for the user and for a common goal [35]. Figure 2.2 is taken from the book [35].

Concurrent systems can be classified as parallel or distributed using the following method. In parallel computing, all processors can have access to a shared memory to exchange information between them [38]. On the other hand for distributed computing, each processor has its own private memory [39]. An example of parallel systems is micro processors with the same memory, whereas an example of distributed systems is any servers in one system that have separate memories for each one. Like in Figure 2.3, part (A) is an example of a distributed system while part (B) is an example of a parallel system.

Some references like [40–43] have no differentiations between distributed computing and parallel computing. The same system could be considered both as “parallel” and “distributed”;

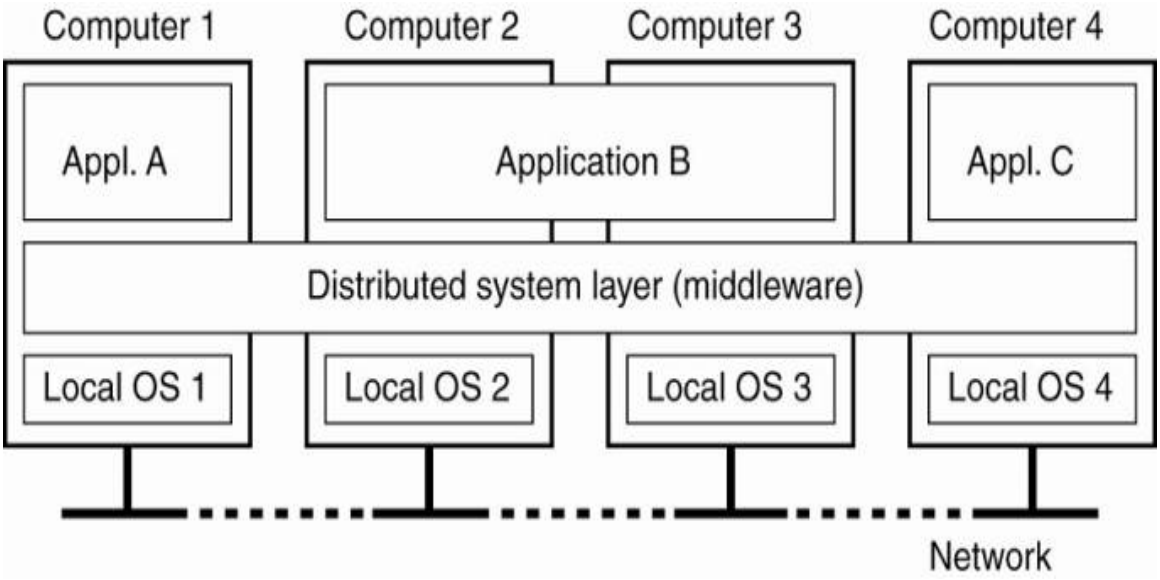


Figure 2.2: A distributed system example

the processors in a typical distributed system run concurrently in parallel [40]. Parallel computing may be seen as a particular form of distributed computing, and distributed computing might be seen as a form of parallel computing. Further, as explained above a number of references have no differentiations between them. Finally and according to cloud designers, they could choose how to implement their systems.

Short historical part of distributed systems is in this and following paragraphs. The first description of the social interactions that could be enabled through networking was a series of memos by Licklider of MIT (Massachusetts Institute of Technology) in 1962 [44]. Later in 1965, working with Merrill, Roberts connected the TX-2 computer in Mass [45]. Moreover, the first widespread distributed systems were local-area networks such as Ethernet which was invented in the 1970s [46]. In late 1966, Roberts went to Defense Advanced Research Projects Agency (DARPA) [47] to develop the computer network concept and quickly put together his plan for the Advanced Research Projects Agency Network (ARPANET), publishing it in 1967 [48]. To make it clear, ARPANET is probably the earliest example of a large-scale distributed application [49]. Distributed systems appeared as early as the Internet has started.

As in [50], the term “Distributed Systems” may be used in autonomous processes that run on the same physical computer and interact with each other by message passing. As it is known of cloud computing systems, one physical computer in provider side can host thousands of VMs [51, 52].

A distributed system could have a common goal, like solving a large computational prob-

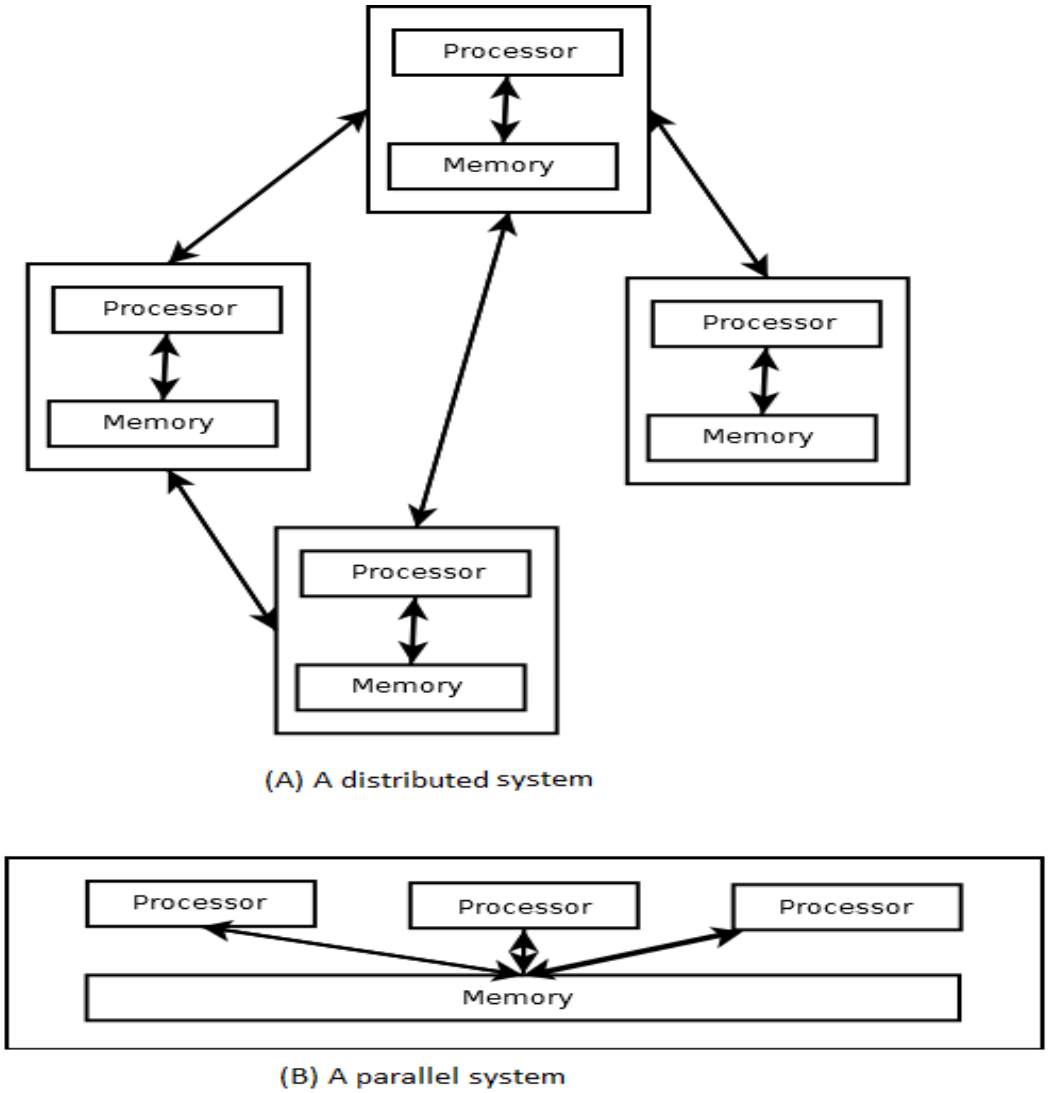


Figure 2.3: Distributed systems and Parallel systems

lem [42]. One of the functionalities of a cloud system, as an example of distributed systems, is distributed monitoring of Virtual Machines in order to detect intrusion by discovering their symptoms.

## 2.4 Cloud computing

This thesis concentrates on cloud computing systems and some of their functionalities. Cloud computing is used everywhere [53] by most of the users of the Internet. There are a number of reasons for making this claim, such as cloud computing can be used everywhere at any time

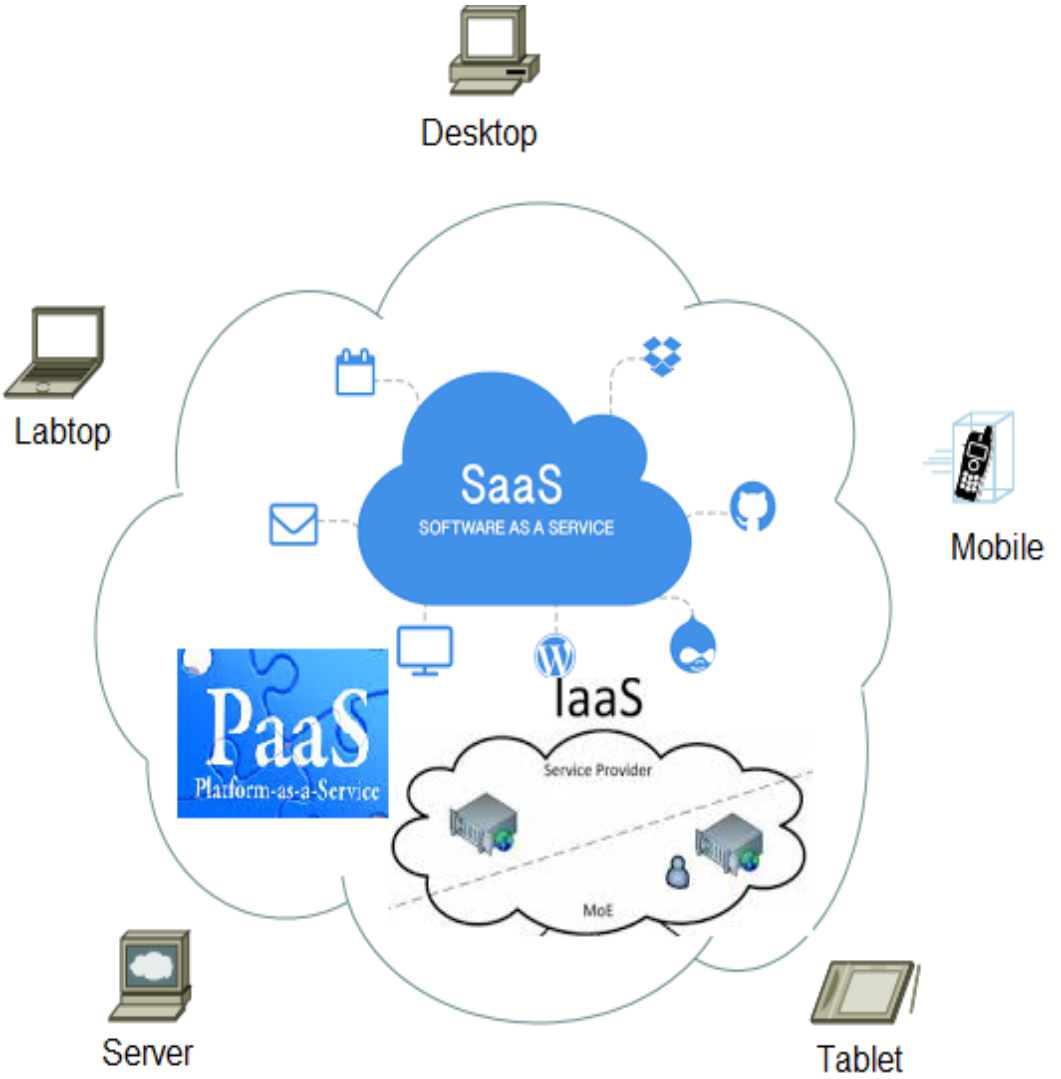


Figure 2.4: Cloud computing high-level architecture

using any device [54], cloud computing does not need high-quality equipment from user side [55], and it is also easy for sharing data to anybody in anywhere in the world [56].

**Definition 2.3.** [57, 58] Cloud computing is a kind of Internet-based computing that gives shared processing resources and data to computers and other devices on demand. It is a model for enabling everywhere, on-demand access to a shared pool of configurable computing resources such as networks, servers, storage, applications and services.

From this definition, it can be seen how beneficial cloud computing is and why it is used widely. Cloud computing concept usually refers to two aspects. The first one comprises applications and services over the Internet. The second refers to the hardware and software

components in data centres offering respective services [11]. Hence, the purpose of cloud systems is to improve the performance of data centres by merits their virtual services. Those services can be accessed from anywhere in the world on demand upon users' requirements (Quality of Service) [59].

To understand cloud better, Figure 2.4 shows that a connection to a cloud system could be by using any device. In addition, this figure presents the three service models of cloud computing as follows Software as a Service (SaaS); in which cloud customers release their applications on a hosting environment [15]. The second one is Platform as a Service (PaaS); PaaS is a development platform supporting the full Web-based application-development [60]. Finally, Infrastructure as a Service (IaaS); cloud consumers directly use the raw IT infrastructures (processing, storage, networks, and other fundamental computing resources) that are available in the cloud system [61].

Some of cloud history is needed here to understand where the cloud computing term came from. Cloud computing is not a new term for the development of web applications [62]. The term "Cloud Computing" is based on a collection of many old and few new concepts in several research fields like Service-Oriented Architecture, distributed and grid computing and visualisation [63].

Some references to Cloud Computing in its modern sense appeared as early as 1996 [64]. The popularisation of the term can be traced to 2006, when Amazon.com introduced its Elastic Compute Cloud [65]. Then, a number of other companies started offering a cloud system, like Google App Engine (GAE) [66], Microsoft Azure [67] and IBM blue Cloud [68].

To explore clouds and the work in this thesis, this section is divided to three subsections. First of them is about the security of clouds and how dangerous cybercrime is. This is to know how important this research and other works in the area are. The second subsection is about cloud monitoring, as this work focuses on it. Final subsection is about cloud structures.

### 2.4.1 Cloud security

In this new era of life, where technology is used from the early morning to late night, cybercrime is becoming more developed and thus challenging for the system designers. The reason is reflected in the increased number of ways used by criminals. The exact definition of cybercrime is not only including on-line crime or computer-related crime [69]. The form of illegal activity changes in time, as cybercriminals change their methods continuously [70]. Payment

card fraud on-line is estimated at 210 million US dollars in the United Kingdom, and this includes only losses from the banks [71]. These losses cause 14 per cent of the UK customers to avoid purchases over the Internet [71].

There are two types of cyber-crime [72]. First of them is that computer is the tool of the crime [73]. This could include child pornography, fraud and criminal harassment. The second type is that computer is the object of the crime [74]. This type includes hacking or unauthorised use of computer systems and the creation of computer viruses. This type is that the one is considered here in this thesis, especially in the context of cloud computing.

As [75] indicates, cloud-computing systems are easy targets for cybercriminals, who are constantly looking for vulnerabilities in the system to explore, due to the distributed nature of clouds. The same paper [75] also argues that intrusion could be detected by behaviour analysis. Here are some of the security issues in cloud computing. The cloud system architecture of the software that is used to deliver cloud services, typically involves multiple intercommunicating cloud components [76]. This intercommunication media could be vulnerable to attacks by outsiders. One of the security issues is related to cybercriminals' attacks. Multitenancy in VM-based cloud infrastructures in the way physical resources are shared among guests, VMs could get a high possibility of an attack [76]. For instance, the way cloud was developed could give an attacker the ability to have administrative control of guest VMs during a live connection by employing a man-in-the-middle attack to modify the code used for authentication [77].

Another type of cloud data security problems is relevant to VM images. A VM image encompasses the software stack, including installed and configured applications, used to boot the VM [76]. An attacker could be able to examine images to determine whether they leak information or provide an avenue for attack [78]. Furthermore, one of security drawbacks is cloud availability. It means that the provider has the full set of computing resources accessible and usable at all times [76]. However, some cases of outage took place with several companies: Amazon's Simple Storage Service (S3) and EC2 services suffered a three-hour outage that, in turn, affected Twitter and other startup companies using the Amazon's services in February 2008 [79]. Not only Amazon's cloud services were affected, but also a database cluster failure at Salesforce.com caused an outage for several hours in February 2008 and in January 2009 [80]. In addition to these outages, in March 2009, Microsoft's Azure cloud service experienced severe degradation for about 22 hours due to networking problems [81].

Finally, it is required to have a good and secure cloud system. It could be done through a number of processes. The most important of them is that to have a good cloud monitoring.

In the next subsection, it will be discussed.

### 2.4.2 Cloud monitoring

Cloud monitoring is important to maintain high system availability and performance for both providers and users [82]. Cloud monitoring helps to scale resources and make the best utilisation of cloud systems [83]. In the same reference [83], it was mentioned that there is a strong need for efficient monitoring looking at the nature of cloud computing.

The problem with existing cloud monitoring is that a considerable amount of basic computing of cloud systems does not subdivide nicely into small tasks [84]. Here in our work, the task of monitoring VMs is subdivided among Forensic Virtual Machines. In order to monitor cloud systems, our approach attempts to carry out monitoring processes in a distributed fashion reflecting the distributed nature of cloud systems.

In regards to current monitoring systems of cloud infrastructures, they are based on local, centralised or hierarchical model approaches [85]. ArcSight Enterprise Security Manager works in a centralised way [85]. The authors of this paper also mentioned other monitoring systems like Cloudkick, Zenoss, Amazon Cloudwatch and CloudSec that work in a centralised model. Some others using hierarchical structure such as HASBE. There are a number of companies that could help the work in network monitoring. For example, Zabbix [86] is a general purpose company that has open source distributed monitoring solution for networks and applications that can be customised for the use of cloud monitoring.

In this thesis, two main distributed monitoring structures of cloud systems are used. The first structure is homogeneous way of distributing VMs. The second one is a hierarchical distribution. For these reasons, the next subsection describes structures of cloud systems, discusses these ways of structuring and gives some examples.

### 2.4.3 Cloud structures

In this subsection, there is an explanation of some previous research in cloud computing that uses different cloud structures to solve the considered research problems. Cloud systems could be designed according to cloud providers and users needs [87]. However, other different reasons could control the design structure of cloud computing. For example, security aspects

should be considered in cloud structures [88, 89]. Also, a designed and optimised structure of data centres is needed [55]. Next paragraphs describe some research that uses different cloud structures.

Most previous studies on scheduling are conducted on homogeneous computing systems [90]. The authors of the papers [90–95] used homogeneous structure of their models. Here in this thesis, most of the algorithms experiments are done in the homogeneous structure of VMs. However, there are different structures as mentioned above. In [96] the authors used the hierarchical structure which is called HASBE for their system. The reason for them to use this structure is to enhance security: each party is associated with a public key and a private key, with the latter being kept secretly by the party. The trusted authority acts as the root of trust and authorises the top-level domain authorities [96].

Furthermore, Ganglia [97] uses a hierarchical designed distributed monitoring system for cluster and other high performance computing systems. This way of structuring is used in this thesis. Another structure is used in [98]. This structure is an end-to-end cloud network architecture that utilises BRITE topology for modelling link bandwidth and associated latencies. Finally, in designing the experiments, used structures are hierarchical and homogeneous systems. In Chapter 6, there will be a discussion about them in the context of this work.

## 2.5 Symptom discovery

The origin of this work comes when Keith Harrison got a US Patent in 2010 [99] for inventing a model of detecting a threat. This was performed by observing multiple behaviours of a system in program execution from outside of a host virtual machine. The processes include mapping a portion of the physical memory of the system to a FVM. The reason is to specify the presence of a first signature of the threat (symptom).

**Definition 2.4.** [18] A symptom is an abstraction of some characteristic that can be related to malicious behaviour, so that occurrence of a symptom shows possibility of a malicious behaviour.

**Definition 2.5.** [18, 99] Forensic Virtual Machines are tiny virtual machines that monitor other VMs in order to identify symptoms. The job of FVMs is to recognise the first indication of a threat that could be a virus or other type of malware or misbehaviour.

After that, Keith Harrison and his co-authors published the paper titled “A framework for detecting malware in cloud by identifying symptoms” [18]. The paper inspired the author of this thesis to start the work it in September 2013. It proposes Mobility Algorithm, which is the most relevant tool for discovering symptoms.

The start of this research is from the idea of this seminal work [18]. The authors claimed that depending on parameterisation, the Mobility Algorithm could be more or less efficient in cloud monitoring. The thesis address this conjecture. It proposes and analyses a variant of Mobility Algorithm, that is called Reduce-Max Algorithm, for monitoring purpose.

The main idea of cloud monitoring is to be able to frequently inspect VMs by a number of FVMs. The target as described is to detect symptoms of malicious behaviours. Section 2.5.1 gives examples of symptoms that could be discovered as signatures of a threat. To be informed, one threat could result in more than one symptom, and a symptom could be related to different threats.

### 2.5.1 Examples of symptoms

- Server overheating [100–102]: this symptom could be related to a malicious behaviour. However, from given references, sometimes overheating is not a symptoms of a threat, but if it comes with other symptoms, it gives an impression of a threat behind.
- Modification of in-memory code: malwares usually make sure that the system continues with its normal behaviour after injecting a code for attack into the in-memory code. To discover that, checking code regularly is one of the processes of discovering symptoms [18].
- Modifying the time attributes of a file. One of the malicious symptoms is related to any change in file attributes [16].
- Packets with unexpected TCP acknowledgment settings. If there is an ACK-flag set through a packet and no previous SYN-packet has been sent. This fact could be associated with an intrusion attempt. Such a situation might also be a symptom of packet damage, a malfunctioning network of software elements, and not an attack attempt [17].
- Missing processes: it is a crucial technique of malware to remain hidden as long as possible. Malware used it to stop crucial processes which might help in its detection [18].

- Service mix attributes. Usually, it is possible to define a standard set of inbound and/or outbound services that are given to a specific user. For example, if the user is on a business trip, he is expected to use email and file transfer options only. Any attempt, on his account, via Telnet to access various ports may indicate an intrusion [17].
- Identifying suspicious snippets of code: use of crypto algorithms is very popular with malware writers. Snippets of program code that has been obfuscated, or the code containing known crypto algorithms, can be a sign of malicious behaviour [18].

# Part II

# Algorithms and Methodology

# Chapter 3

# Detection Algorithms

## 3.1 Overview

This chapter presents the objectives of this work and the algorithms that are used for achieving these objectives. As explained in the first part of this thesis, the aim of this thesis is to design and evaluate new algorithms for performing distributed monitoring of cloud systems. This chapter provides definitions and descriptions of all algorithms.

This chapter is organised as follows: Section 3.2 provides important definitions that are used for all algorithms. This section is followed by Section 3.3 that presents all data sets that are used later in the experiments' chapters. Section 3.4 is about algorithms themselves. The algorithm section is divided into six subsections as follows. Section 3.4.1 is about the general Reduce-Max algorithm. Section 3.4.2 contains definitions of the two versions of Reduce-Max. After that, Section 3.4.4 contains a model about the hierarchical system and the dependence on clusters for Reduce-Max algorithm. Section 3.4.5 describes Random-Start-Round-Robin algorithm. Section 3.4.6 is about balancing Mobility algorithm. Section 3.5 presents the objectives of this research. The reason of stating the objectives is to know what the proposed algorithms are about.

## 3.2 Problem definitions

The aim of this thesis is to design and evaluate new algorithms for performing distributed monitoring of cloud systems. The main functional objective is to minimise the latencies of

visiting VMs. First, let us describe the meaning of the following elements. The start is with VMs that can be located in physical machines. One physical machine could have thousand of VMs. FVMs are Forensic Virtual Machines, which are small virtual machines (VMs) that monitor other VMs. The design algorithms based on Mobility algorithm. Next is a description of Mobility algorithm.

FVMs are typically autonomous simplified VMs with limited computability and communication. This concept was more formally defined in the context of cloud security in [18, 99], where Mobility Algorithm was defined for general purpose of searching for symptoms and discovering their dangerous configurations. The Mobility Algorithm has become an important tool for cloud monitoring providing lightway security, though it is difficult to analyse due to its complexity. Its simplified version, called Reduce-Max Algorithm, was formally specified and analysed in more depth in [2]. The aim of the Reduce-Max Algorithm was only to monitor the system by assuring frequent visits to VMs for identification of symptoms, without taking into consideration additional criteria imposed by security objectives. Two structures of VMs are studied in this thesis. First one is homogeneous system. VMs in these systems are in one pool. The second one is hierarchical structures. VMs in hierarchical systems distributed among clusters.

In this thesis $k$ refers to the total number of configurations, each configuration of symptoms has a number of related symptoms. For example, one configuration has 4 symptoms. $Disc(c_i, v)$ is the number of symptoms in one configuration $c_i$ discovered at VM $v$. While $size(c_i)$ is the number of potential symptoms in configuration $c_i$, $val(c_i)$ is an assigned value saying how dangerous the configuration $c_i$ is. $\lambda(v)$ is a vector of weights that has a value associated with delay at VM $v$ (also called a *weight* of $v$) and $T(v)$ is the current time minus the last time when VM $v$ was visited by a FVM. $N$ references the number of VMs and $M$ is the number of FVMs. For weighting VMs, $VM_i$ denote the $i$-th VM. Finally, Equation in (3.1) is to assess the urgency of visiting a Virtual Machine $v$

$$f(v) = \sum_{i=0}^{k} \frac{Disc(c_i, v)}{size(c_i)} val(c_i) + \lambda(v)T(v), \tag{3.1}$$

A further definition is about the total-max. Total-Max is the highest weighted latency ($\lambda(v)T(v)$) between two visits to a VM that can be observed from the whole system. The problem is to schedule the visit of FVMs to VMs. The main functional objective is to minimise the latencies of visiting VMs. This means minimising Total-Max.

## 3.3 Data sets

In this section, a specification of all data sets for distribution weights among VMs is given.

### 3.3.1 Uniform weights

The start of uniform distribution concept was in Hermann Weyl paper in 1916 [103]. Then, a sequence is called to be uniformly distributed if all values are equal [104].

Uniform inputs in this thesis mean that all VMs are equal in regards to weights $\lambda$; without loss of generality, it could be assumed that all of the weights of VMs are ones [3].

### 3.3.2 Random weights

Random inputs for weighting VMs in this thesis means that the values of weights are selected randomly and independently from integer values from a given range. We use two different ranges for Random weights. The first one is Random [1,3] weights. The range of it includes the values of 1, 2 and 3. The second one is Random [1,15] weights, which includes the random selection of 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14 and 15. The reason of choosing these two different random ranges is that firstly is to have small values of random weights to compare the total-max of these weights with both total-max with bigger random range [1-15] and with Poisson (2). Secondly, We will also compare the range [1,15] with Poisson (8).

### 3.3.3 Poisson weights

The Poisson distribution is introduced by a French mathematician, Simeon Denis Poisson [105]. It is a discrete probability distribution that specifies a probability of a given number of events occurring in a fixed time or space [106]. Poisson distribution is used here in this thesis. The reason of using Poisson distribution is to have two different random weights which are more complex and widely applicable random distribution. Poisson distribution, which is based on parameter $\mu$, generates values for the weights of VMs.

To have selected random choices apart from just two range of Random weights, two different $\mu$ are used here. The first one is $\mu = 2$ for Poisson (2) weights. The range of the

random selection will be around 2. The second one is $\mu = 8$ for Poisson (8) weights. The range here also is around 8.

### 3.3.4 Arithmetic integer sequence weights

Arithmetic integer sequences of weighting VMs means that the weight of the $i$th VM equals to $i$. For instance, weight of the first VM, equals to 1, weight of the second VM equals to 2 and so on until the weight of the $N$-th VM equals to $N$.

### 3.3.5 Harmonic weights

Harmonic numbers for weighting VMs are such that the weight of the $i$th VM equals to $\frac{1}{i}$, i.e., $\frac{1}{1}$ for $VM_1$, $\frac{1}{2}$ for $VM_2$, $\frac{1}{3}$ for $VM_3$, ... $\frac{1}{i}$ for $VM_i$ ... and the $\frac{1}{N}$ for $VM_N$.

### 3.3.6 Exponential weights

The values of exponential weights for weighting VMs are such that the weight of the $i$th VM equals to $\frac{1}{2^i}$, i.e., $\frac{1}{1}$ for $VM_1$, $\frac{1}{2}$ for $VM_2$, $\frac{1}{4}$ for $VM_3$, ...,$\frac{1}{2^i}$ for $VM_i$ ... and the $\frac{1}{2^N}$ for $VM_N$.

## 3.4 Algorithms

All algorithms are going to be explained in this section.

### 3.4.1 Reduce-Max algorithm

**Definition 3.1.** Reduce-Max Algorithm is a simplified version of the Mobility algorithm [18]. In this algorithm, a decision about the next visited VM $v$ is made by considering the (highest) value of a simplified function $f(v) = \lambda(v)T(v)$, then the $T(v)$ of the visited VMs is reduced to 0 that causes the weighted latency becoming 0.

**Algorithm 1** Reduce-Max algorithm

**Input:** $n$ = number of VMs, $r$ = number of rounds, $v$ = a specific VM
**Output:** TotalMax

```
1:  n = 1024
2:  r = 100000
3:  TotalMax = 0
4:  index = 0                                   ▷ to find the highest latency
5:  loop
6:      for j = 0 to j = r do                   ▷ MAIN LOOP: do this for every round j < r
7:          for v = 0 to v = n do               ▷ Adding values to v
8:              array(v) = array(v) + v)
9:          end for
10:         for v = 0 to v = n do               ▷ to find the highest latency
11:             if array(v) > array(index) then
12:                 index = v
13:             end if
14:         end for
15:         if TotalMax < array(index) then
16:             TotalMax = array(index)         ▷ The highest latency
17:         end if
18:         array(index) = 0                    ▷ Reduce highest latency
19:     end for
20: end loop
```

To understand the definition of Reduce-Max algorithm, a pseudo code in Algorithm 1 is given.

At the beginning of the thesis work, this algorithm was analysed for eight different weighting configurations of VMs: Uniform, Arithmetic, Harmonic, Exponential, (uniformly) Random [1,3] and [1-15] weights and Poisson (2) and (8) weights. To understand Reduce-Max better, next paragraphs contain more discussions about two of these weighting configurations.

- Uniform weights

  Recall that Uniform weights mean that all VMs are considered to have the same weighting latency. It implies that $N$ VMs can be visited once in $N$ rounds. This is because all VMs have the same weights. For example, all of the VMs' weights are 1. The following fact illustrates Reduce-Max in the case of Uniform weights.

  **Fact 1.** Suppose $\lambda(v) = 1$ for every VM $v$. Then the highest weighted latency observed in any execution of the Reduce-Max algorithm is at most $N$ for visiting all VMs.

  *Proof.* We prove that the following invariant holds in every round. Invariant: for every $1x \geq 1$, the number of VMs of latency at least $x$ before each reduce operation is at most $N - x + 1$. Now we prove the invariant by induction on the number of reductions. Before the first reduction, all weights are 1, therefore the invariant is clearly satisfied. Assume that the invariant holds before reduction $t \geq 1$ and we will prove that it also holds before reduction $t + 1$. In reduction $t$ the highest latency is reduced to 0, while all other latencies increase by 1. The former operation combined with the invariant means that after it, for every $x \geq 1$, the number of VMs of latency at least $x$ before each reduce operation is at most $(N - x + 1) - 1 = N - x$. The latter operation implies that all VMs with latency at least $x - 1$ now have latency at least $x$, so the number of such machines is at most $N - (x - 1) = N - x + 1$, which completes the proof of the invariant. It follows from the invariant, satisfied before each reduction (as we just showed), the number of VMs of latency at least $N + 1$ is at most $N - (N + 1) + 1 = 0$, therefore there is no VM with latency higher than $N$. □

- Arithmetic integer sequence weights

  Recall that Arithmetic integer sequences of weighting VMs means that the weight of the first VM equals 1, the weight of the second VM equals 2 and so on until the weight of the $N$-th VM equals $N$.

**Fact 2.** Suppose $1 \leq \lambda(v) \leq N$, $\lambda(v) = v$ for $v = 1, 2, \ldots, N$. Then the highest weighted latency observed in execution of algorithm Reduce-Max is at most $N \cdot (N-1)$.

*Proof.* Suppose, to the contrary, that at some point the maximum latency exceeds $N(N-1)$. Let $t$ be the first such moment of reduction and $v$ denote the VM with the maximum latency at that time. Consider this reduction together with $N$ preceding reductions - among them there must be two reductions done on the same VM, say $w$, by pigeonhole principle (applied to $N$ different VMs and $N+1$ reductions). There are two cases.
Case 1: $v = w$. It means that the number of increases of latency of $v$ between the two consecutive reductions of its latency is at most $N-1$, which means that the reduction at time $t$ was on latency at most $N \cdot (N-1)$ (recall that the highest increase is $N$). This contradicts the initial assumption and thus proves the theorem in this case.
Case 2: $v \neq w$. Consider the last time $t'$ before $t$ when the latency of $w$ was reduced. Note that this reduction took place when latency of $w$ was at most $\lambda(w) \cdot (t' - (t-N) - 1) \leq N \cdot (t' - (t-N) - 1)$. At that time, the latency of $v$ could not be bigger than that number, by the principle of reducing the highest latency. Since that time, the latency of $v$ has increased by at most $N \cdot (t - t' - 1)$. Therefore, the latency of $v$ at time $t$ could be at most $N \cdot (t' - (t-N) - 1) + N \cdot (t - t') = N(N-1)$, which is a contradiction with the initial assumption and thus completes the proof in this case and the whole theorem. □

### 3.4.2 Deterministic-Centralised and Randomised-Local versions

Here in this subsection, we have two versions of the original Reduce-Max algorithm. First one is Deterministic-Centralised version. The second one is Randomised-Local version of Reduce-Max Algorithm.

**Definition 3.2.** Deterministic-Centralised version means that the $i$-th FVM reduces the $i$-th highest weighted latency from VMs.

**Definition 3.3.** Randomised-Local version (also called distributed) means that each FVM reduces a randomly selected VM from some range of the VMs with highest weighted latencies.

The pseudo code for Deterministic-Centralised version is in Algorithm 2 and The pseudo code for Randomised-Local version is in Algorithm 3.

The focus is on a comparison of local randomised algorithms that patrol the cloud system independently, with a coordinated (deterministic) algorithm in which the patrolling FVMs

**Algorithm 2** Deterministic-Centralised algorithm

**Input:** $n$ = number of VMs, $m$ = number of FVMs, $r$ = number of rounds, $v$ = a specific VM
**Output:** TotalMax

```
1:  n = 1024
2:  m = 16
3:  r = 100000
4:  TotalMax = 0
5:  index = 0                                   ▷ to find the highest latency
6:  loop
7:      for j = 0 to j = r do                   ▷ MAIN LOOP: do this for every round j < r
8:          for i = 0 to i = m do               ▷ For FVMs
9:              for v = 0 to v = n do           ▷ Adding values to v
10:                 array(v) = array(v) + v)
11:             end for
12:             for v = 0 to v = n do           ▷ to find the highest latency
13:                 if array(v) > array(index) then
14:                     index = v
15:                 end if
16:             end for
17:             if TotalMax < array(index) then
18:                 TotalMax = array(index)     ▷ The highest latency
19:             end if
20:             array(index) = 0                ▷ Reduce highest latency
21:         end for
22:     end for
23: end loop
```

**Algorithm 3** Randomised-Local algorithm

**Input:** $n$ = number of VMs, $m$ = number of FVMs, $r$ = number of rounds, $v$ = a specific VM, *range* = the top latencies.
**Output:** TotalMax

```
1: n = 1024
2: m = 16
3: r = 100000
4: top = 16
5: TotalMax = 0
6: index = 0                                          ▷ to find the highest latency
7: loop
8:     for j = 0 to j = r do                          ▷ MAIN LOOP: do this for every round j < r
9:         for i = 0 to i = m do                      ▷ For FVMs
10:            for v = 0 to v = n do                  ▷ Adding values to v
11:                array(v) = array(v) + v)
12:            end for
13:            for v = 0 to v = n do                  ▷ to find the highest latency
14:                if array(v) > array(index) then
15:                    index = v
16:                end if
17:            end for
18:            for z = 1 to z = top do                ▷ For the top latencies
19:                if range[z] ≠ array[index] then
20:                    range[z] = array[index]
21:                end if
22:            end for
23:            if TotalMax < array(index) then
24:                TotalMax = array(index)            ▷ The highest latency
25:            end if
26:            random = a random number from range [z].
27:            array(random) = 0                      ▷ Reduce highest latency
28:        end for
29:    end for
30: end loop
```

cooperate on selecting the most urgent targets. The former uses fewer resources but may impose additional cost (e.g., smaller frequency of monitoring some VMs) due to the lack of coordination. In this thesis, we assess this overhead and discuss when and which variant of local randomised algorithm to use, i.e., what range of selection to choose.

### 3.4.3 Symptoms-appearance

This section of the thesis suggest the using of Reduce-Max Algorithm under the assumption of symptoms appearance. It explains the work conducted by the author on monitoring Virtual Machines with the target of discovering malicious symptoms.The focus is on a comparison of results of Randomised algorithms that patrol the cloud system independently with the same sets of the algorithm after the appearance of symptoms. The appearance of symptoms in VMs will be fixed.

### 3.4.4 Hierarchical topology and dependence on the number of clusters

#### 3.4.4.1 Introduction

Cloud customers could use VMs as services, platforms or as a whole infrastructure. However, in practice the infrastructure of a computing cloud includes several levels, such as virtual gateways, virtual clusters and virtual nodes. In this section, we pursue a study of the impact of a hierarchical structure, formed of three levels, on the process of monitoring the system, with the main goal of discovering symptoms of malicious behaviours in clouds. This thesis addresses, in the model of hierarchical systems, two major questions. The first question refers to optimising the number of clusters in the hierarchical structure to guarantee efficient monitoring. The second question, posed in some previous papers [2, 4, 6] in this area, concerns efficient distributed implementation of the monitoring process; namely, how to choose locally the next VM to be visited by a FVM from specific range of highest VMs.

#### 3.4.4.2 Detail of the work on the impact of hierarchical structure

Cloud computing systems can be studied on different levels. Firstly, virtual gateways could be defined as software run by a cloud operator to make cloud resources available to the users

[107]. A cloud gateway secures user data and allows cloud providers to better control their data [108]. Virtual gateway in this concept could contain a number of virtual clusters. Virtual cluster comprises of a number of VMs that are configured to be consistent on one cluster to share physical resources [109]. The main goal of this part is to study how the hierarchical structure of a cloud influences the efficiency of the monitoring process, and how to implement it in an efficient, light and local way.

The objective function we want to minimise is the maximum value $f(v)$ associated with any VM $v$ during the execution; we call it also a *total-max*. This corresponds minimisation of the maximum possible accumulated weight associated with a VM in the process of waiting for a visit of some FVM.

In this part, we study the performance of the Randomised-Local reduction version of Reduce-Max. This version is particularly important because it requires little resources (i.e., is lightweight) and no coordination. Our simulations are performed in hierarchical tree system, which models well the real structure of the cloud.

Two sets of weighting VMs are used here and were already studied in [3]: uniform and random. The reason is that for these sets of weight the monitoring was most efficient in homogeneous systems that studied earlier for Reduce-Max algorithm and its two versions, therefore it is natural to study them in the hierarchical system. Poisson distribution is also used from the paper in [4] in the context of homogeneous systems which is also studied for balancing Mobility Algorithm. It is a more complex and widely applicable random distribution.

The performance of monitoring is measured as the maximum waiting latency time (or, alternatively, as the maximum accumulated weight since the last visit of a FVM), depends on the number of clusters and the ranges of random selection of highest VMs. The impact of the former is bigger as in Figures 5.24, 5.26, 5.28, 5.30 and 5.32 in Section 5.5 in Chapter 5, and can burst for wrongly chosen number of clusters, while the impact of the latter is less but in general it is preferable to set up ranges bigger than the number of exploring FVMs. This confirms how important proper structure of the cloud is for efficiency of the monitoring process, in particular, for implementing lightweight security and resiliency systems on the top of the cloud.

The idea is to have different number of clusters. FVMs and VMs will be distributed among clusters. The work of FVMs for each cluster will be separated from others. Finally, a comparison between different number of clusters to examine the effectiveness of hierarchical structure.

### 3.4.5 Random-Start-Round-Robin algorithm

In this subsection, there is another algorithm for the work of this thesis, which is a Random-Start-Round-Robin algorithm. For Reduce-Max algorithm, the reduction was from the highest weighted VMs (even with its two versions, the reduction either of the exact highest weighted VMs or from a range of the highest values).

Now with Random-Start-Round-Robin algorithm, the way of reduction is different. One of VMs is selected randomly by one of FVMs. After that this specific FVM reduces the weight of the selected VM wherever its position and it is not necessary it has the highest weight. After that, this FVM keep reducing the latencies of all VMs in round robin fashion. All other FVMs follow the same procedures.

### 3.4.6 Balancing Mobility algorithm for monitoring Virtual Machines

Forensic Virtual Machines (FVMs) choose the best possible VM to inspect. Each FVM carries a copy of an algorithm (e.g., Mobility algorithm) to know the next target Virtual Machine. It is crucial to make it distributed to suit the nature of cloud systems. Harrison et al. in [18] state that it is not possible to deploy many FVMs for each VM. They declare this because of the vast scale of the cloud system [110] with limited resources available for security aspects [111]. This does not contrast with the fact of virtually unlimited computational resources of cloud computing. A cloud service-provider offers numerous resources for cloud customers and users [112]. However, these resources are for users and should not be wasted on cloud security. Therefore ideally, FVMs should use small resources and avoid costly coordination and centralised computing.

In this section, the proposed approach is to balance the two parts of the mobility Formula 3.1, that is proposed in [18]. It is by setting weights before each part of the mobility Formula as in 3.2. The total of the two weights equals to one ($W1+W2=1$). If the first weight before symptom part equals to zero and the second one before second part equals to one, then the algorithm becomes the Reduce-Max Algorithm. If Mobility algorithm is balanced as above, then it can be found the fastest way of visiting VMs, regardless of the appearance of symptoms.

$$f(v) = W1 \cdot \sum_{i=0}^{k} \frac{Disc(c_i, v)}{size(c_i)} val(c_i) + W2 \cdot \lambda(v) \cdot T(v), \tag{3.2}$$

## 3.5 Objectives

The main objective of all algorithms is to perform a distributed monitoring of cloud systems for intrusion detection. Therefore, the main question was about what are the most efficient algorithms for visiting VMs with the goal of detecting symptoms of intrusions. The main functional objective is to minimise the latencies of visiting VMs.

The objectives have a number of related issues as described in Chapter 1. In the following points, more details about the related objectives are provided:

- How beneficial Reduce-Max algorithm and its two versions are?

  Reduce-Max algorithm will be investigated in Section 4.2.1 and the results will be in Section 5.2. The two versions are Randomised-Local reduction and Deterministic-Centralised reduction. In this chapter, both versions are defined.

- How much are the differences between the best results of Randomised-Local reduction and the results of Deterministic-Centralised reduction?

  The reason of this objective is to see the importance of each version of reduction especially Randomised-Local reduction that has great benefits toward saving time in coordination and communication. The two versions are modelled in Section 4.2.2. Their results are in Section 5.3.

- How does the Random-Start-Round-Robin algorithm differ from the two versions of the Reduce-Max algorithm?

  The model of Random-Start-Round-Robin algorithm is in Section 4.2.5 and the results are in Section 5.6 and finally a discussion about it in Section 6.2.2.

- What is the best structure of cloud networks with respect to efficient monitoring?

  Two cloud structures are used in this thesis. They are hierarchical and homogeneous structures. The model of hierarchical structure is in Section 4.2.4 and the results are in Section 5.5 and finally a discussion about it in Section 6.2.3.

- What are the most efficient ranges of highest weighted VMs for Randomised-Local reduction in both cloud structures?

  They are used in both homogeneous and hierarchical structures. The discussion about it in Section 6.4.

- Moreover, there is a question about how Reduce-Max algorithm behaves with appearance of symptoms.

  The model of the appearance of symptoms is in Section 4.2.3 and the results are in Section 5.4.

Finally, more challenges cloud be regarding data sets and their suitability for each number of VMs and each algorithm. However, cloud designers could choose data setting that suits their systems according to the number of FVMs, architecture and appropriate structures. Next is the

# Chapter 4

# Methodology of Experiments

## 4.1 Overview

This chapter of part two of this thesis is named as the methodology of experiments for all algorithms proposed in the previous chapter. After this introductory section, an evaluation environment of each algorithm is provided. At the end of this chapter, a summary is presented in Section 4.3.

## 4.2 Evaluation environment of each algorithm

Table 4.1, shows the settings that are used to test the algorithms.

Table 4.1: The settings used to test the algorithms

| Algorithms | no. of FVMs | no. of VMs | No. of Rounds | No. of Clusters |
|---|---|---|---|---|
| Reduce-Max (Original) | 1 | From 5 to 1024 | 100000 | 1 |
| Two versions of Reduce-Max | From 1 to 16 | 1024 | 100000 | 1 |
| Symptoms Appearance | From 1 to 16 | 1024 | 100000 | 1 |
| Hierarchical structure | 16 | 1024 | 100000 | From 1 to 16 |
| Random-Start-Round-Robin | 16 | 1024 | 500000 | From 1 to 16 |
| Balancing Mobility-algorithm | From 1 to 16 | 1024 | 100000 | 1 |

### 4.2.1 Reduce-Max algorithm

It was assumed at the beginning of this thesis that the design system just has one type of FVM, and all VMs are connected. Because of this assumption, $M$ equals 1, $Disc(c_i, v)$ should be equal 1 when the symptom has been discovered, or 0 when there are no symptoms.

Additionally, it is assumed that there is a complete network of $N$ VMs, also called nodes, and just one type of FVMs. Each VM $v$ has its own weight ($\lambda_v$). Reduce-Max algorithm schedules FVM to visit VM $v$ with the maximum value of $\lambda_v \cdot T_v$, where $T_v$ is the latency time from the last visit of some FVM to VM $v$. Recall that weighted latency of VM $v$ at round $r$ is defined as $\lambda_v \cdot T_v$, where $T_v$ is taken at round $r$. More precisely, $T_v$ is a function of round $r : t_v = t_v(r)$. If the round $r$ is clear from the context, then we simply use $T_v$. We call $T_v(r)$ the latency.

In this algorithm, a decision about the next visited VM $v$ is made by considering the (highest) value of a simplified function $f(v) = \lambda(v) \cdot T_v(r)$. After a FVM visits to this specific VM ($v$), the weighted latency reduced to zero.

This algorithm at the beginning of this work uses eight different weighting configurations as described in chapter three: (uniformly) Random ([1,3] and [1,15]), Uniform, Poisson (2), Poisson (8), Arithmetic, Harmonic, and Exponential.

Simulations are done for algorithm Reduce-Max. $N$ is between 5 and 1024 VMs. The range of round $r$ is from 1 to 100000.

### 4.2.2 Deterministic-Centralised version and Randomised-Local version of Reduce-Max algorithm

The two versions of Reduce-Max for more than one FVM uses five different weighting configurations: (uniformly) Random [1,15], Uniform, Arithmetic, Harmonic, and Exponential.

There are two methods of reduction for Reduce-Max algorithm. The first one is *Deterministic Coordinated reduction*, which means that the $i$-th FVM reduces the latency of the $i$-th highest weight of VMs. For instance, if the system has two FVMs, together they will reduce the first two highest weights of VMs.

The second way is *Randomised-Local reduction*. In the Randomised-Local reduction, each FVM chooses randomly from a given range of highest weights. The considered ranges of highest values are: 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 and 16.

In each execution of the program, we consider 1 to 16 FVMs. These FVMs reduce the highest weights of 1024 VMs by both methods of reduction. That is because of the assumption of one host that contains 1024 VMs.

Furthermore, a comparison is based between the best result of the Randomised-Local reduction, for each number of FVMs, and the Deterministic-Coordinated reduction and analyse the differences. The best result is selected among the algorithms using different highest ranges, from 2 to 16. If the differences that occur are small, we will conclude that it is better to use Randomised-Local reduction of a specific highest range rather than the Deterministic Coordinated one, because it saves time as we choose the next VMs locally from just a range of highest values and do not spend resources on coordination. This is much more efficient than finding, by the Deterministic-Coordinated algorithm the highest weights of a VM over all the VMs' weights, as the latter requires coordination mechanisms between FVMs.

### 4.2.3 Symptoms-appearance with Reduce-Max algorithm

Section 4.2.2, the work with Mobility algorithm was with the assumption of zero symptom. This assumption makes the first part of the formula unused. Here in this section, we have the assumption of symptoms appearance to analyse how important is the appearance of symptoms. It is assumed that we have one configuration of symptoms that contains four symptoms. These symptoms occur randomly in VMs.

Simulations are done for algorithm Reduce-Max, which reduces the highest value of VMs' weights. The reduction used here is the Randomised-Local reduction. Each FVM chooses randomly from a given range of highest weights, and the considered ranges of highest values are 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 and 16.

As explained above, an assumption is about a random appearance of symptoms in VMs. The number of symptoms is 4 in one configuration, which means $K = 1$. For example, if 2 symptoms have appeared in a VM in round $i-1$, $Disc(c_i, v) = 2$ and $size(c_i) = 4$. Then, the calculation will use mobility algorithm and Reduce-Max algorithm to see which VM should be visited.

Simply, in this section, a calculation is based on the best result of the Randomised-Local reduction, for each number of FVMs (from 1 to 16) and compare that with the result of the Randomised-Local reduction in Chapter 5 to analyse the differences. The best result is selected among the algorithms using different highest ranges, from 2 to 16.

### 4.2.4 Hierarchical topology and dependence on the number of clusters

In this section a tree structure of a cloud is considered. This structure includes three levels: gateway level, clusters level and nodes level.

The method of reduction studied in this setting is the Randomised-Local Reduction, as it is a lightweight implementation and has been studied in previous papers [3], [6] and [4]. The results of Deterministic Coordinated Reduction are also used as optimal results to be compared with the best results of the Randomised-Local reduction for each number of cluster.

What distinguishes this setting from the previous sections is that this time the Reduce-Max algorithm is studied for a different architecture — the hierarchical tree structure. This structure contains three levels. The first level is a gateway level. Then the second level is the level of clusters. VMs are distributed among these clusters on the third level. We assume that there is one physical host. The host has 1024 VMs in total, organised into the tree. The considered number of VMs is motivated by the fact that in [113] and [114] the authors state that one host could have up to 1024 VMs. This host has one virtual gateway including a number of clusters. In the experiments, the number of clusters vary from 2, 4, 8 and 16. It includes only even numbers. However, there is a special case of just one cluster which means FVMs check VMs as in a single cluster for comparison. VMs are divided among these clusters in a uniform way, that is, every cluster has the same number of VMs. The weights of VMs inside each cluster are normalised to sum up to one, which corresponds to the case when clusters are similar in terms of monitoring demands and manage them independently.

Clusters control FVMs that visit and inspect VMs inside the clusters. This means that a cluster creates and kills FVMs according to the needs. In this way, resources will be saved for customer needs. However, in the simulations, the number of FVMs is fixed to test the effectiveness of hierarchical structure. 16 FVMs are divided among clusters. The number of 16 FVMs was used in [3], [6] and [4]. Another parameter considered in this work is the range of highest values of $f(v)$, from which a FVM selects randomly a VM to visit. This is modelled in simulations by a parameter $\alpha$, where $\alpha$ = 1, 1.2, 1.4, 1.6, 1.8, 2, 2.2, 2.4, 2.6, 2.8, 3. The

value of the range of highest latencies comes from multiplying the total number of FVMs by the value of $\alpha$.

As explained above, the system will have three sets of data for weighting VMs, which have been used in paper [3] and [4]. The sets are Uniform, Random distribution and Poisson distribution of weights. For Poisson and Random, the values of implementation differ according to the expected value $\mu$. Two values of $\mu$ are considered and they are 2 and 8. For Random distribution, the ranges of randomness are based on two random sets, they are random integers from 1 to 3 and random integers from 1 to 15.

For each setting of parameters, each execution has been run for 100000 rounds and compute the total-max. After that, for each experiment, the execution repeated 100 times and take the average; this way done due to the randomness of the results in order to eliminate random fluctuation of the results. The results will be presented in the next chapter.

### 4.2.5 Random-Start-Round-Robin algorithm

In this part, we consider Random-Start-Round-Robin algorithm that is different from the previously described ones. Here, we use the results of each two versions of Reduce-Max algorithm to have a comparison with the new algorithm result. What distinguishes this algorithm from the previous ones is that this time the reduction is as follows. FVMs choose one VM randomly in the first round then reduce all VMs in a round robin way.

The assumption is fixed from previous section. There is one physical host. The host has 1024 VMs in total, organised into the tree. Data sets are also fixed and they are Uniform, Random distribution for two different ranges and Poisson distribution of weights for two different parameters $\mu$. For each data set, the results will be for 1, 2, 4, 8 and 16 Clusters. For each setting of parameters, each execution has been run for 500000 rounds and compute the total-max. After that, for each experiment, the execution repeated 100 times and take the average; This done due to the randomness of the results in order to eliminate random fluctuation of the results. The results will be presented in Chapter 5.

### 4.2.6 Balancing Mobility algorithm for monitoring Virtual Machines

The proposed approach is to balance the two parts of the mobility Formula 3.1, that is proposed in [18]. The two weights in Simulations are as follows ($W1 = 0$ and $W2 = 1$).

Simulations are done for the algorithm Reduce-Max, which reduces the highest value of VMs' weights by the fact that some FVM visits this VM.

The method of reduction used in this section is the Randomised-Local reduction. Using this version of reduction, each FVM chooses randomly from a given range of highest weights. The considered ranges of highest values are 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 and 16. In each execution of the program, the considered number of FVMs is from 1 to 16. These FVMs reduce highest weights of 1024 VMs. The best result of the Randomised-Local reduction, for each number of FVMs is selected among the algorithms using different highest ranges, from 2 to 16.

## 4.3 Summary

The main part of this chapter was about evaluation environments of each algorithm. The setting of evaluation environments of each algorithm were presented. Experiments are done for 1024 VMs and mostly for 16 FVMs. Eight data setting were used for all different algorithms. For the number of rounds, all experiments of algorithms are done for 100000 rounds except for Random-Start-Round-Robin algorithms. The number of rounds for it were 500000. The next chapter will be about the results of these experiments.

# Part III

# Results, Discussions and Conclusions

# Chapter 5

# Results of Experiments

## 5.1 Overview

In this chapter, there will be a presentation all results of experiments done for all algorithms, which were proposed in Chapter 3 and are modelled in Chapter 4.

This chapter has seven sections as follows: Section 5.2 presents the results for Reduce-Max algorithm. Section 5.3 shows the results of Deterministic version and Randomised version of Reduce-Max for many FVMs. Section 5.4 presents the results of symptoms-appearance. It is followed by Section 5.5 about the hierarchical structure. Then, Section 5.6 shows the results of Random-Start-Round-Robin algorithm. After that, Section 5.7 presents the results of Balancing Mobility algorithm. Finally, a summary of this chapter is presented in Section 5.8.

## 5.2 Reduce-Max algorithm

To understand what Reduce-Max algorithm is, the initial results of Reduce-Max will be presented in this section. The expectations are that Uniform weights will present the best results because there will not be a big max. That is because all weights are similar.

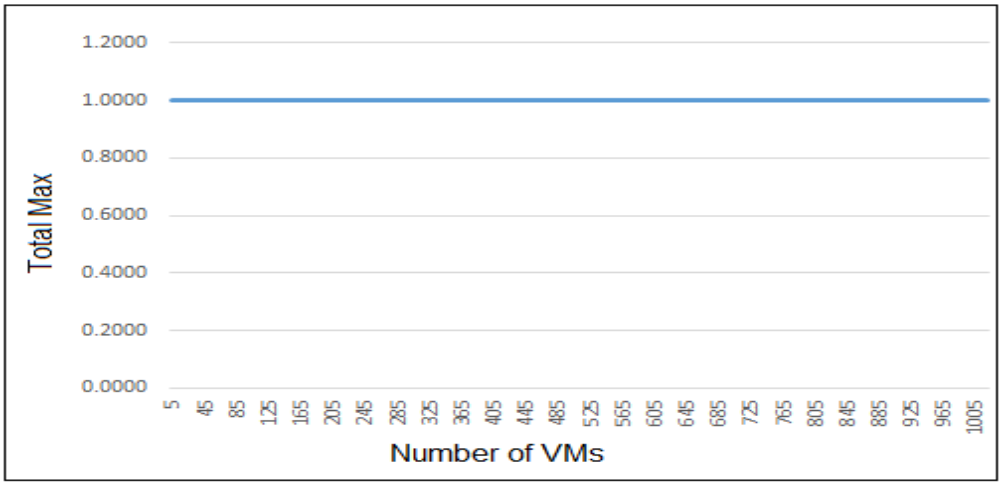


Figure 5.1: Latency trend of normalised total-max for Uniform weights

### 5.2.1 Uniform weights

In Figure 5.1, we see clearly that the normalised total-max for all number of VMs applied is around 1. This is because all VMs have the same weight. Therefore, the reduction of the highest will be in round robin way. For these reasons, all weighted latencies for all VMs are equal. Here why the results in Figure 5.1 are 1s.

### 5.2.2 Random [1,3] weights

Weights here are uniformly random based on a randomly selected weigh t from the integer range that includes $1, 2$ and 3. For instance, weight one for $VM_1$ could be 3 and weight two for $VM_2$ could be 1 and so on. The results in Figure 5.2 are between 1.33 as the biggest value of total-max and the value of 1.2. However, the trend goes smooth around 1.25 with the bigger number of VMs in the system.

### 5.2.3 Random [1,15] weights

Weights here are random by a randomly chosen integer value from 1 to 15 as a weight for each VM. For instance, weight one for $VM_1$ could be 7 and weight two for $VM_2$ could be 13 and so on. The results in Figure 5.3 is just under 1.5 if the number of VMs is only 5 in the cloud system then the trend goes smooth under 1.2 with bigger numbers of VMs. This trend is more

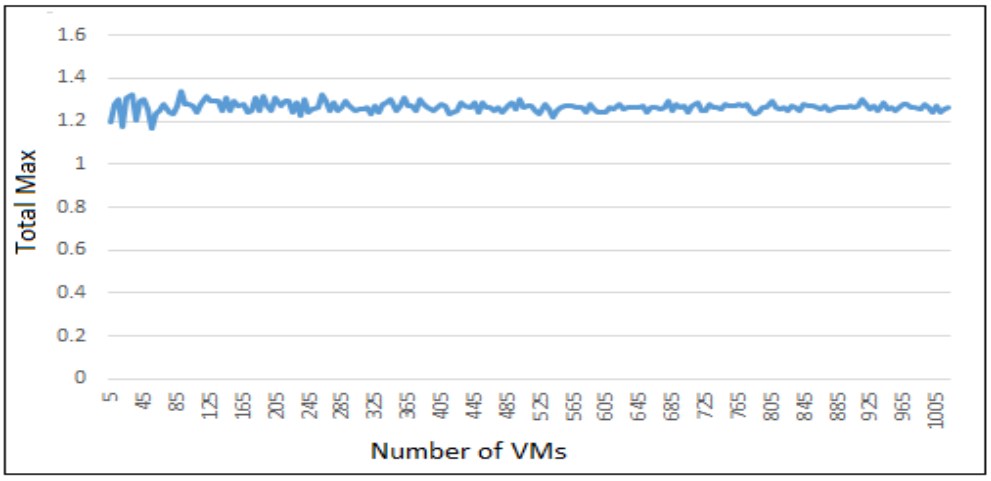


Figure 5.2: Latency trend of normalised total-max for Random [1,3]

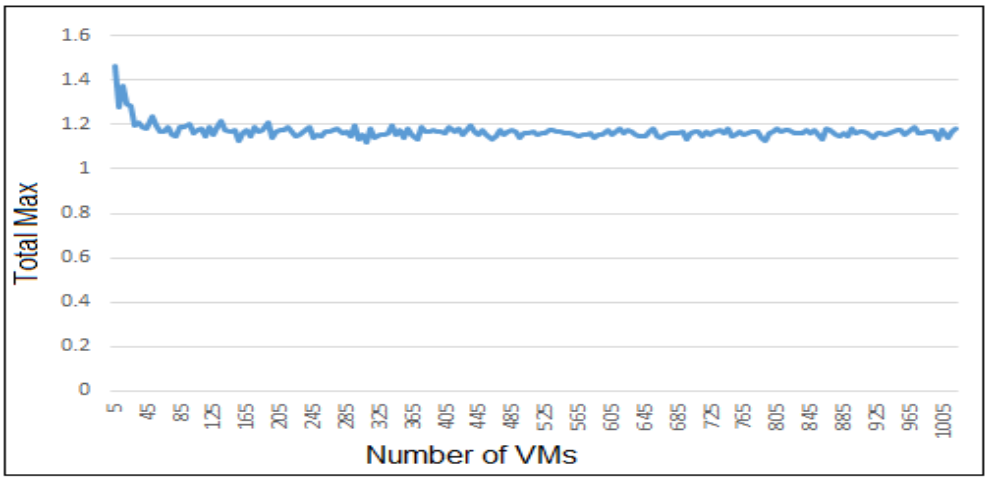


Figure 5.3: Latency trend of normalised total-max for Random [1,15] weights

stable than the one for Random [1,3]. The results of total-max are also better than those for Random [1,3]. The reasons are that for bigger number of VMs, the distribution of randomness selection are bigger and also the possibility of having the same weights is less Therefore, the more distributed randomisation the more stable trend and smaller results.

### 5.2.4 Poisson (2) weights

Weights here are distributively given based on a selection of Poisson randomness that depends on the parameter $\mu = 2$. The results in Figure 5.4 is around 1.5 with the number of 5 VMs. The trend here goes smooth around 1.2 with bigger numbers of VMs. This trend is also stable

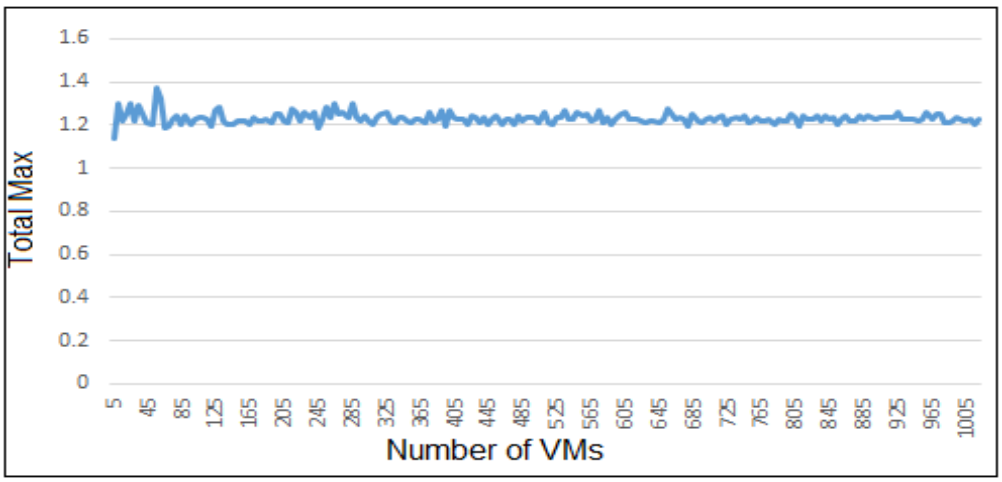


Figure 5.4: Latency trend of normalised total-max for Poisson (2) weights

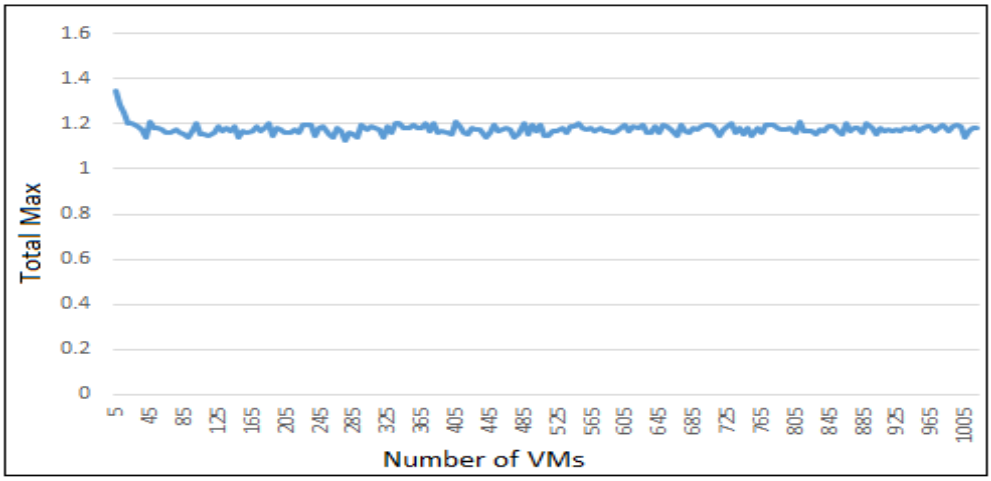


Figure 5.5: Latency trend of normalised total-max for Poisson (8) weights

and results are generally lower than the one for Random [1,3].

### 5.2.5 Poisson (8) weights

Weights here are based on a selection of Poisson distribution for $\mu = 8$. The results in Figure 5.5 is just under 1.3 when the number of VMs is 5 in the cloud system. After that, the trend is between 1.2 and 1.15 with bigger numbers of VMs. This result of 1.15 is the best among all Random sets either they are uniformly random or Poisson random. The trend is stable, which suggests that for Poisson distribution with the growth of expectation the results get generally better.

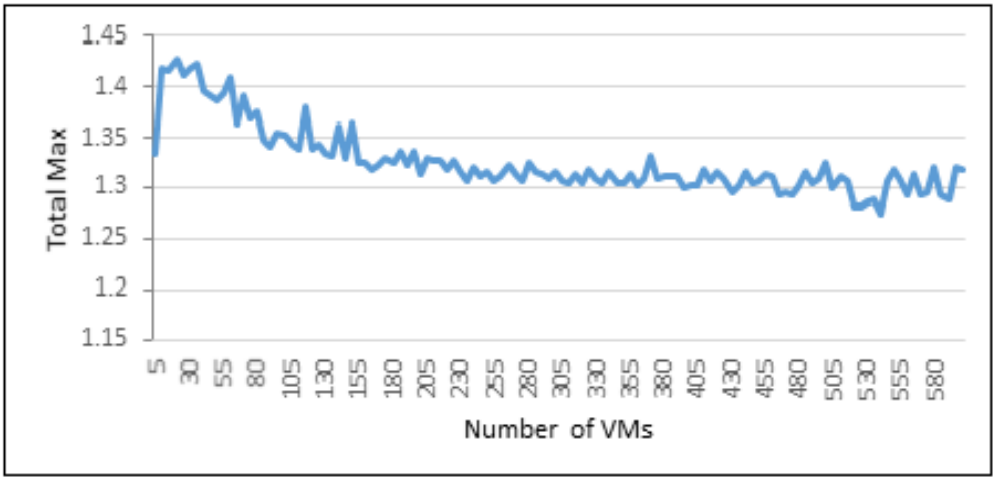


Figure 5.6: Latency trend of normalised total-max for Arithmetic series weights

### 5.2.6 Arithmetic integer series weights

After normalising the total-max for Arithmetic series weights, the results are as shown in Figure 5.6. They start from above 1.4 for any number of VMs that is less than 40 and go down to 1.3 with a growing number of VMs. The trend of normalised total-max is less stable and the results are a bit bigger than other sets of weights. That could be because there is bigger differences between Arithmetic weights. For instance, weight of VM1 equals to 1 and VMN equals to N.

### 5.2.7 Harmonic weights

Harmonic numbers for weighting VMs are from $\frac{1}{1}$ for $VM_1$, $\frac{1}{2}$ for $VM_2$, $\frac{1}{3}$ for $VM_3$, ... $\frac{1}{i}$ for $VM_i$ ... and towards $\frac{1}{N}$ for $VM_N$. It can be shown that for this setting of weights, there is a better trend of weighted latency compared with the arithmetic integer series. Figure 5.7 shows that the trend of normalised total-max latency for Harmonic weights goes down just under 1.2 and is stable, while in arithmetic integer series of weights, it drops to 1.3 as in Figure 5.6. In Figure 5.7, for harmonic weights, the results reach 1.5 for small numbers of VMs and go down to be less than 1.2 with bigger numbers of VMs.

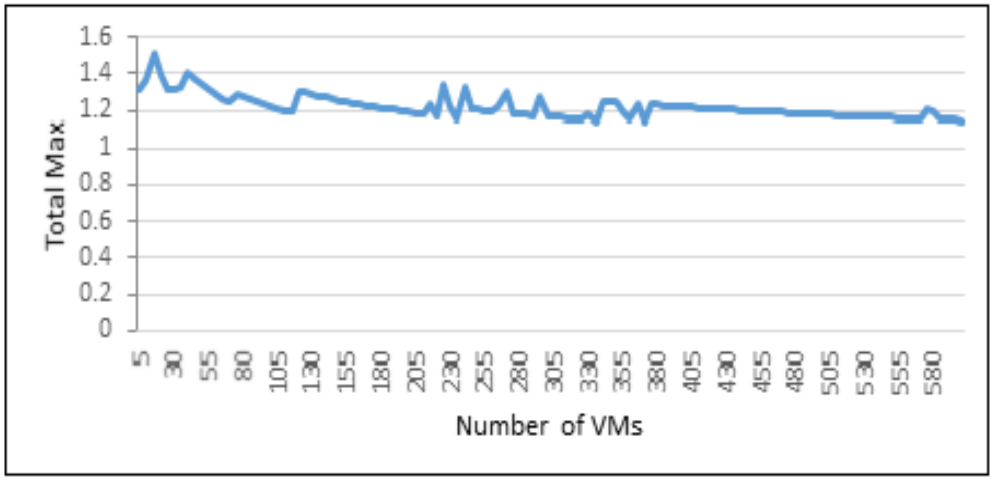


Figure 5.7: Latency trend of normalised total-max for Harmonic weights

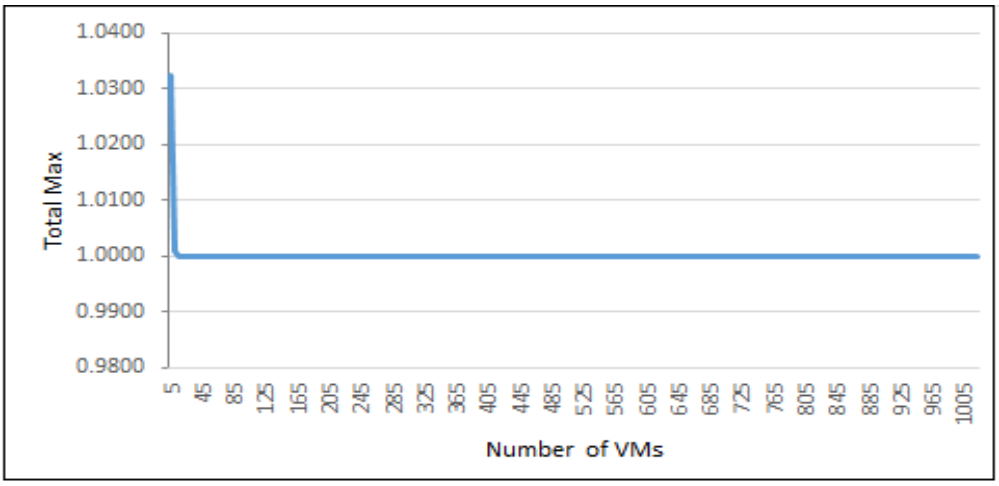


Figure 5.8: Latency trend of normalised total-max for Exponential weights

### 5.2.8 Exponential weights

Values' numbers for weighting VMs here are from $\frac{1}{1}$ for $VM_1$, $\frac{1}{2}$ for $VM_2$, $\frac{1}{4}$ for $VM_3$, ...,$\frac{1}{2^i}$ for $VM_i$ ... and towards $\frac{1}{2^N}$ for $VM_N$. In Figure 5.8, an indication for exponential weights, the results are around 1.03.

The trend in Figure 5.8 is stable and the results are small, but this is probably because of the large number of VMs, almost all weights are very small and only few are large.

### 5.2.9 Conclusion

From previous results, the ones for Uniform weights as expected are considered the best among all of them. The best here means in regards to the total-max. It will be the smallest. The reason is that the weighted latency for all VMs is the same. Poisson (8) weights come after as the second best. The total-max when they applied for most number of VMs is better than the one for Random [1,15] and then for Poisson (2). After that Random [1,3] as the worse of random selection. That could be because the selection of weights is from limited number. Arithmetic integer series weights come after. The reason could be because there is bigger gap between the setting of Arithmetic weights. Other weights like Harmonic and Exponential weights, the trends are stable. This is probably because of the large number of VMs, almost all weights are very small and only few are large.

## 5.3 Deterministic-Centralised version and Randomised-Local version of Reduce-Max for many FVMs

In this section, we give and discuss the results of the simulations that are done for each set of input weights and for the two versions of Reduce-Max algorithms.

In the first type of the resulting graphs below, the rightmost column of mauve boxes displays the maximum-weight of algorithm Reduce-Max, in which FVMs collectively choose different VMs from the first $M$ machines of highest latency, i.e., they use Deterministic Coordinated reduction. The next fifteen columns of different coloured boxes display maximum-weight of Reduce-Max algorithm in which FVMs choose VMs using Randomised-Local reduction from the range of 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 and 16 of highest weight VMs, respectively. Sixteen rows correspond to different numbers of FVMs - from 1 to 16. The red colour boxes - one for each considered number of FVMs - denote the boxes with the smallest maximum weight among the results for the random selection from 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 and 16 highest latency VMs, i.e., indicate the best highest range (i.e., cloumn id) to be used by the Randomised-Local reduction for each considered number of FVMs (row).

Note that the mauve boxes in the rightmost column, i.e., for the Deterministic Coordinated reduction, are by definition not higher than the other corresponding fifteen boxes for the same number of FVMs.

In the second type of the resulting graphs, we display the ratio between the best choice of the range of the highest weights (i.e., the value of the corresponding red coloured box in the preceding figure) and the Deterministic-Coordinated reduction (i.e., the value of the corresponding mauve box in the preceding figure), for all considered numbers of FVMs. This ratio is at least 1, by definition, and expresses how much cost (e.g., frequency of visiting) the system lose when applying Randomised-Local algorithm with random selection of VMs instead of the Deterministic-Coordinated one. Recall that the advantage of the former is its lightweight implementation, i.e., not relying on substantial resources or coordination and communication.

### 5.3.1 Random [1,15] weights

In Figure 5.9, the results obtained for the Random weights show that the more FVMs are applied for Deterministic-Coordinated Reduction, the better total-max can be achieved (cf., the mauve boxes). This is as expected because the more FVMs the less load for each FVM. Moreover, the results in Figure 5.9 also show that the more FVMs we apply for Randomised-Local Reduction, the wider range of highest values achieves the best total-max. The same reason for the load of performance for FVMs. This conclusion comes from the observation that the red box for one FVM indicates the range of two highest values as the best for Randomised-Local Reduction; however, the red box moves towards the range of 11 highest values for 2 FVMs. Finally, the red box is in the range of 16 highest values for 12, 14 and 15 FVMs.

Additionally, from Figure 5.9, for only one FVM the total-max get worse for a wider range of highest values. It is around 1.18 for the window of two highest and just under 1.4 for the window of 16 highest values. This is because the number of VMs in the range increase that leads to increase the load for the FVM. However, these results change with the increase of the number of FVMs. For example, the total-max weight is about 0.6 for the range of two highest values for 16 FVMs and then it goes down to just above 0.12 for the range of 16 highest values. The reason here is that the number of FVMs increased for the same window of highest.

The ratio of the best result of Randomised-Local Reduction compared to Deterministic-Coordinated Reduction result for Random weights in Figure 5.10 is increasing from around one for one FVM to around 1.7 for 16 FVMs. This trend of only a slight increase in the ratio is a positive indication for implementing Randomised-Local Reduction, with properly chosen highest range as indicated by the red boxes in Figure 5.9 in practical systems with Random weights.

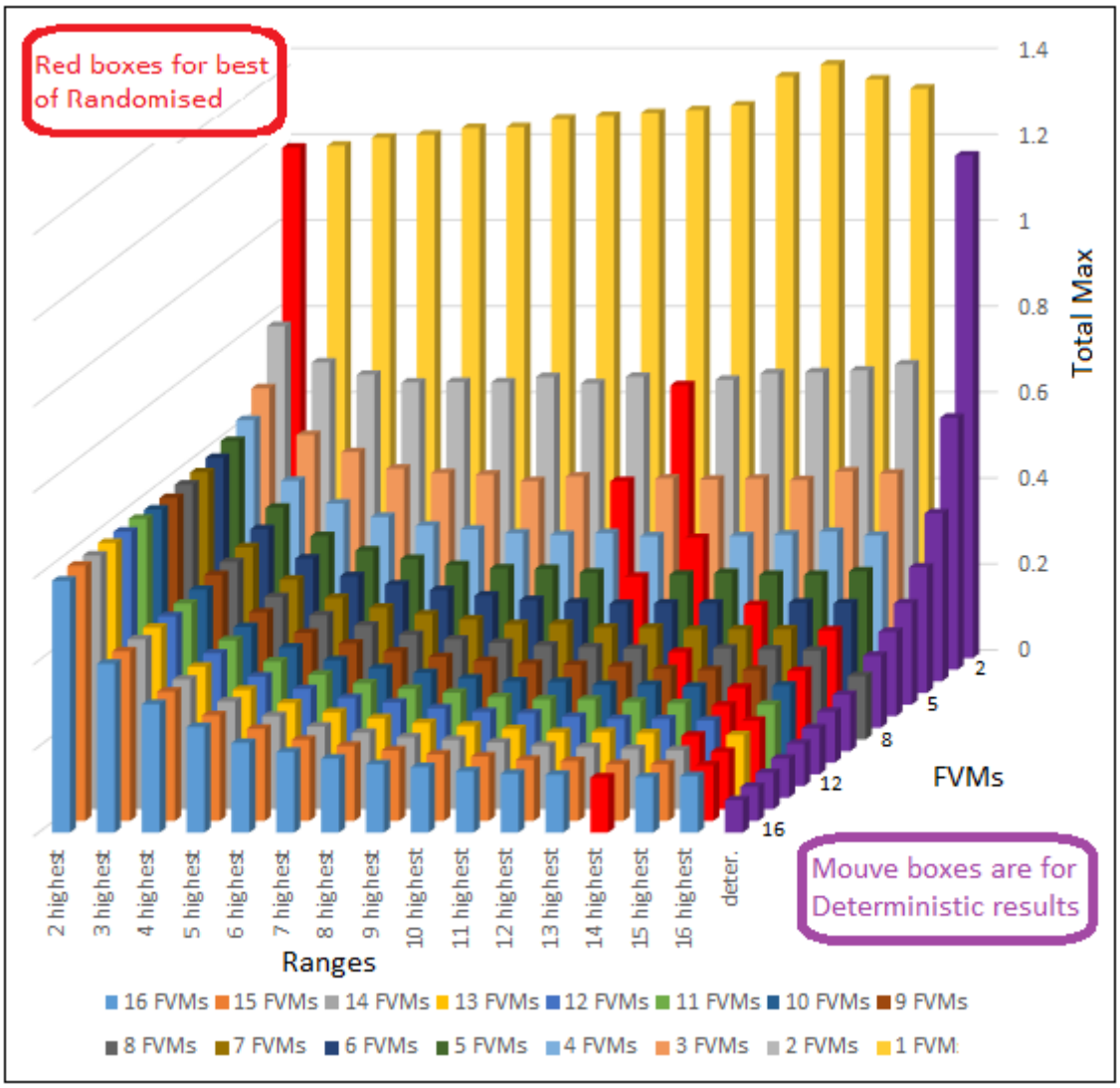


Figure 5.9: The total-max results for Deterministic-Coordinated Reduction and Randomised-Local Reduction, taken for fifteen settings of the highest range, applied to Random weights

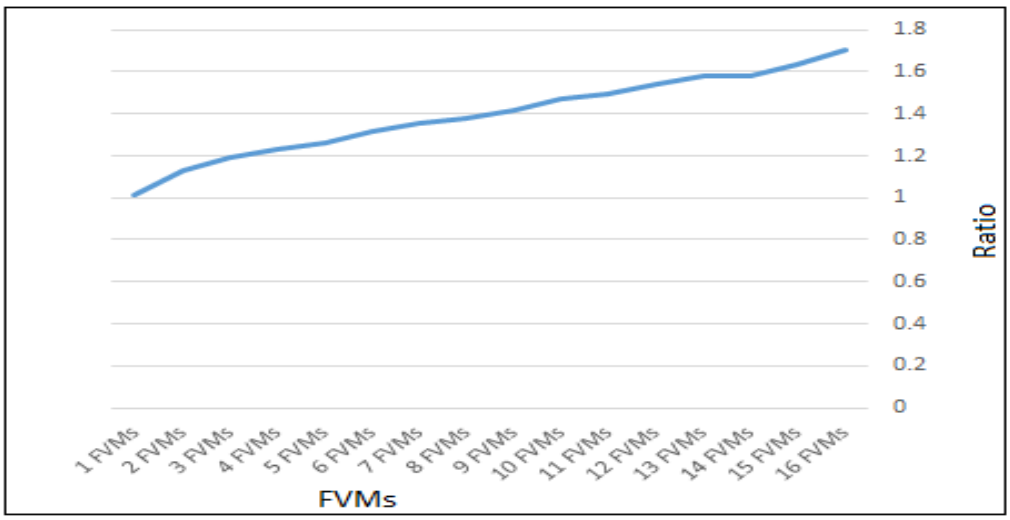


Figure 5.10: The ratio of the best results of Randomised-Local Reduction compared to Deterministic-Centralised Reduction results for Random weights

### 5.3.2 Uniform weights

Similar to random weights, the results of executing Reduce-Max algorithm for Uniform weights in Figure 5.11 indicate that the more FVMs we apply for Deterministic-Coordinated Reduction, the lower total-max we get. As stated, the reason is about that the load is less for a single FVM. In Figure 5.11, the mauve boxes for Deterministic-Coordinated illustrate that the total-max for one FVM is one, for two FVMs is 0.5, for 4 FVMs is 0.25 and it goes down towards 0.0625 for 16 FVMs.

For Uniform weights the red box, that shows the best highest range of each number of FVM, starts from the range of 2. It goes through the ranges of 9 and 13. Finally, it reaches the range of 16 highest latencies for 11, 13, 14, 15 and 16 FVMs. This result met expectation because more FVMs is applied for the same range of highest.

The ratio of the best value of Randomised-Local Reduction compared to the one obtained by Deterministic-Coordinated Reduction for uniform weights, as in Figure 5.12, increases from around 1 for one FVM to just under 1.8 for 16 FVMs. This ratio is similar to the ratio obtained for a random set of weights in Figure 5.10.

The Randomised-Local Reduction is even more efficient for the Uniform weights. The reason is that the curve trend of the ratio between the best result of Randomised Reduction and the Deterministic Reduction result, has some points lower than the ratio for Random weights with the increasing number FVMs..

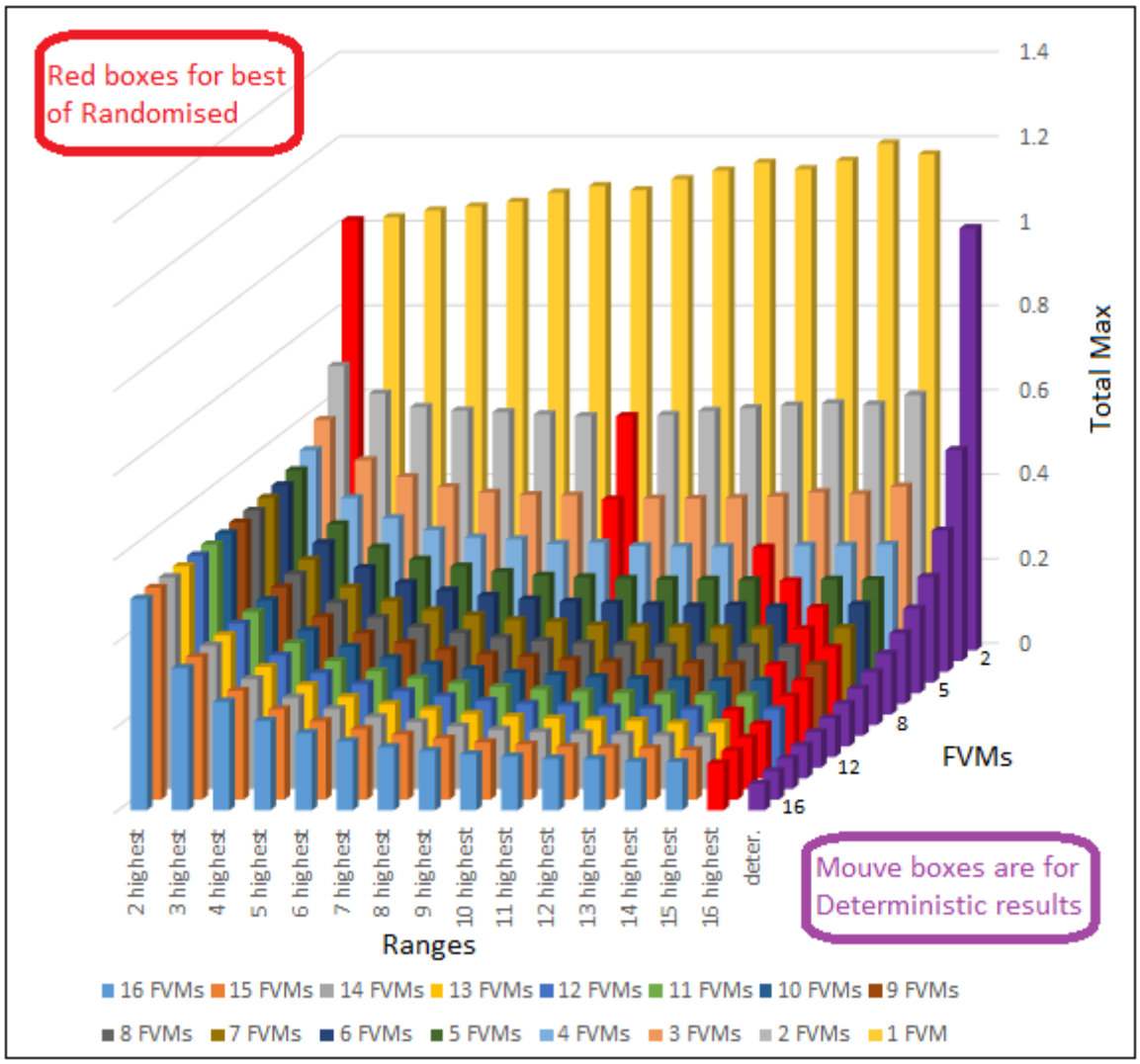


Figure 5.11: The total-max results for Deterministic-Coordinated and for Randomised-Local version of Reduce-Max, taken for fifteen settings of the highest range, applied to Uniform weights

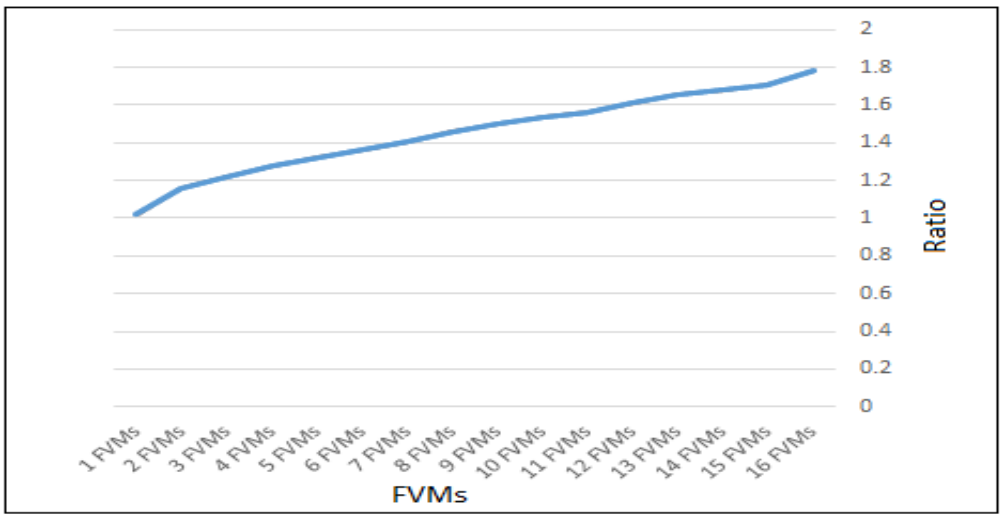


Figure 5.12: The ratio of the best results of Randomised-Local Reduction compared to the results of Deterministic-Coordinated Reduction for Uniform weights

### 5.3.3 Arithmetic integer series weights

The results for Arithmetic integer sequences in Figure 5.13 , illustrate that the decrease in the value of total-max in mauve columns for Deterministic-Coordinated Reduction is similar to the decline of the previously considered weights. It goes from around 1.01 for one FVM towards 0.06 for 16 FVMs.

The pattern of red boxes for choosing the best window of highest values for Randomised-Local Reduction, cf., Figure 5.13, is different from the one in Figure 5.9 for Random weights. It moves from the range of 2 highest through the range of 5, 7, 8 and 12 towards 16 highest. The difference is that here for Arithmetic weights, it stays in smaller ranges such as 5, 7, 8 highest for 2, 3, 4 FVMs. However, it was in the range of 9 highest for 2 FVMs in the case of Random weights.

The ratio of the best value of Randomised-Local Reduction compared to Deterministic-Coordinated Reduction, when applied to the Arithmetic series of weights, cf., in Figure 5.14, is similar to the ratio for Random weights in Figures 5.10 and 5.12. However, ratios in Figures 5.10 and 5.12 are slightly better than the ratio in Figure 5.14. The reason of this statement is that the curve of the ratio makes some results of FVMs bigger. For examples, The ratio is about 1.4 when 5 FVMs applied in Figure 5.14, whereas it is around 1.3 when the same number of FVMs applied.

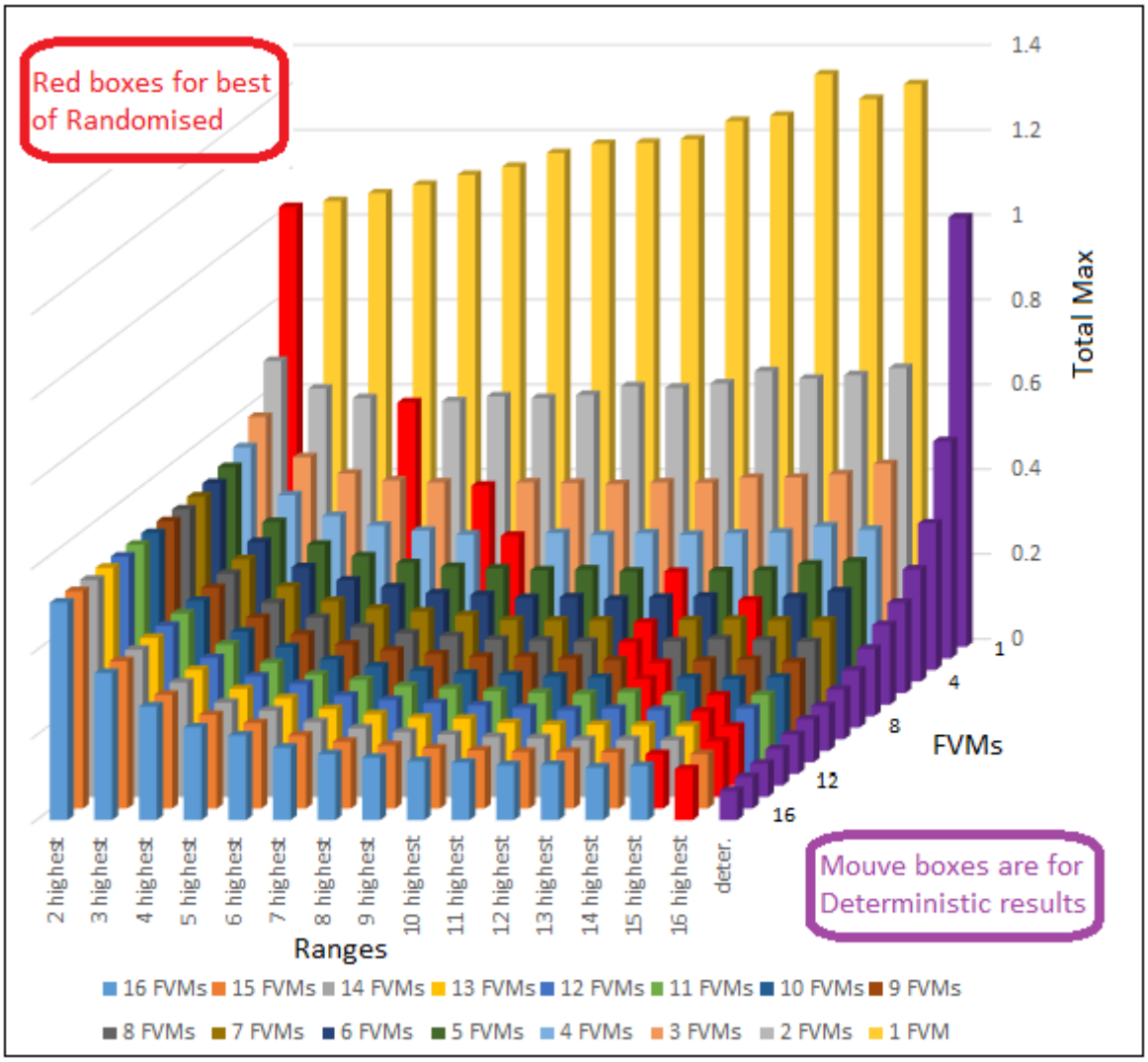


Figure 5.13: The total-max results for Deterministic-Coordinated version of Reduce-Max algorithm and for Randomised-Local version of Reduce-Max, taken for fifteen settings of the highest range, applied to Arithmetic series weights

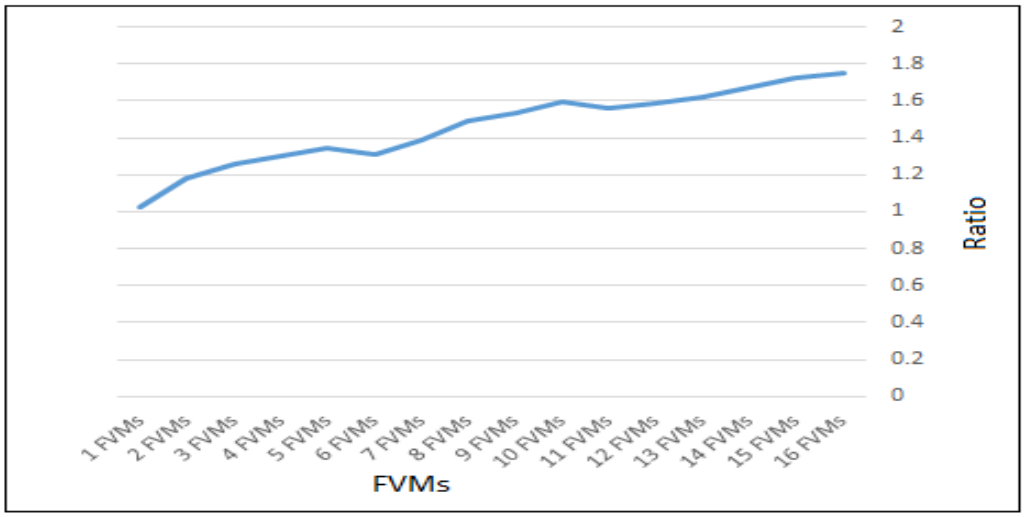


Figure 5.14: The ratio of the best result of Randomised-Local Reduction compared to the results of Deterministic-Coordinated Reduction for Arithmetic weights

### 5.3.4 Harmonic weights

In the case of Harmonic weights, the results in Figure 5.15 indicate that a decline in values of total-max weights in the mauve boxes, corresponding to the Deterministic-Coordinated Reduction, occurs from 1 to 7 FVMs, and then it becomes stable for more FVMs. The reason is about that there is more FVMs for the same number of VMs.

The red boxes of choosing the best window of highest values for Randomised-Local Reduction, cf., Figure 5.15, is in the range of two highest latencies through all numbers of FVMs. Then, it moves towards the range of 3 highest when 11 FVMs are applied, and stays there also for 12 FVMs before it moves again to the range of 4 highest for 14 and 16 FVMs. This is probably because of the large number of VMs, almost all weights are very small and only few are large.

The ratio of the best value of Randomised-Local Reduction compared to Deterministic-Coordinated Reduction for Harmonic weights, as displayed in Figure 5.16, is worse than any previously considered set of weights. However, it reaches the peak for 8 FVMs and then goes down for more FVMs. This trend suggests that we may get better results with more FVMs. However, the best value goes from around 2 for 1 FVM to be just under 6 for 8 FVMs and goes down towards 4 for 16 FVMs that gives worse results. This ratio is different from previous ratios because weights are very small and only few are large.

There are three issues in the case of Harmonic weights. The first problem is that total-

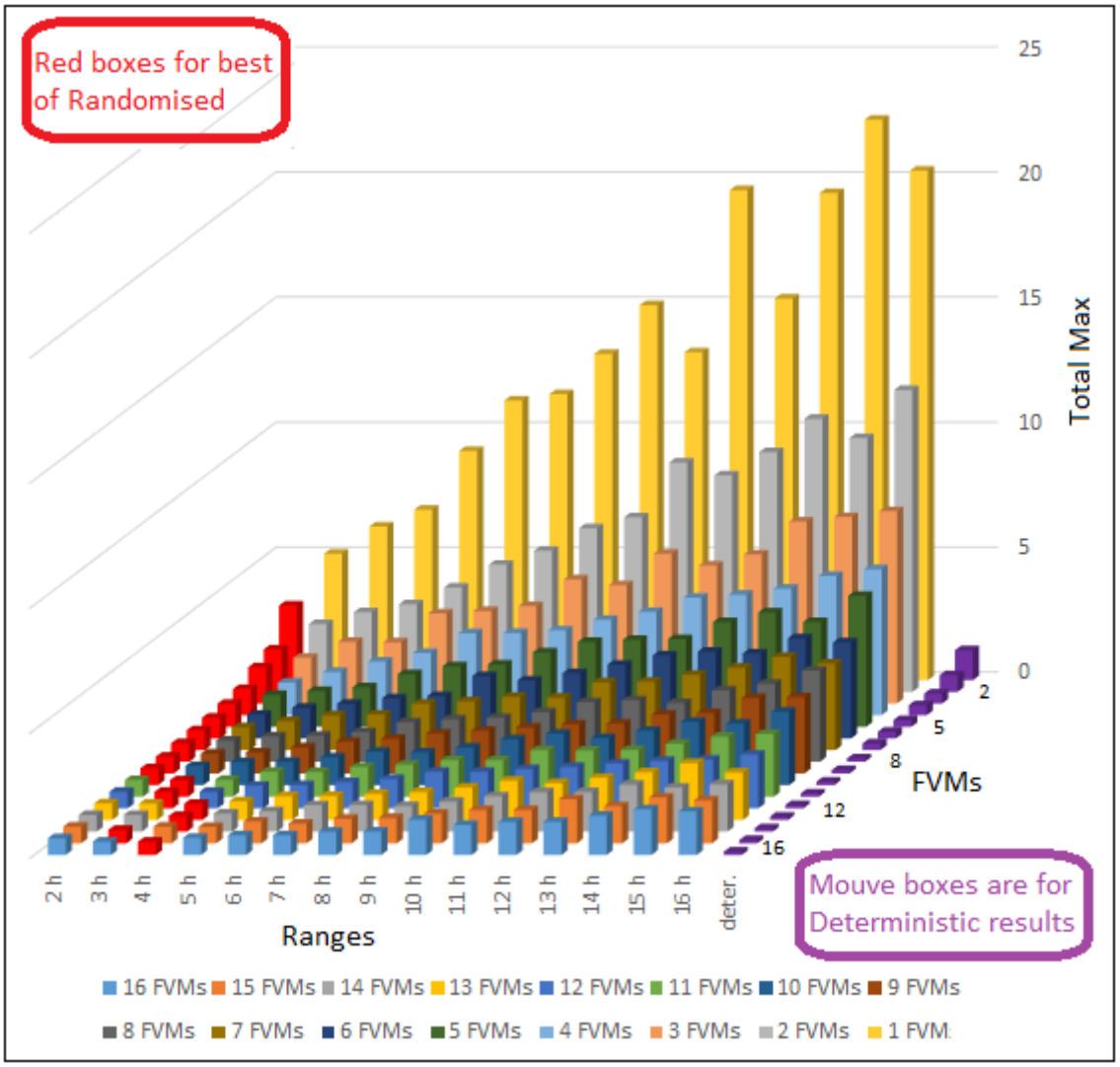


Figure 5.15: The total-max results for Deterministic-Coordinated version of Reduce-Max algorithm and for Randomised-Local version of Reduce-Max, taken for fifteen settings of the highest range, applied to Harmonic weights

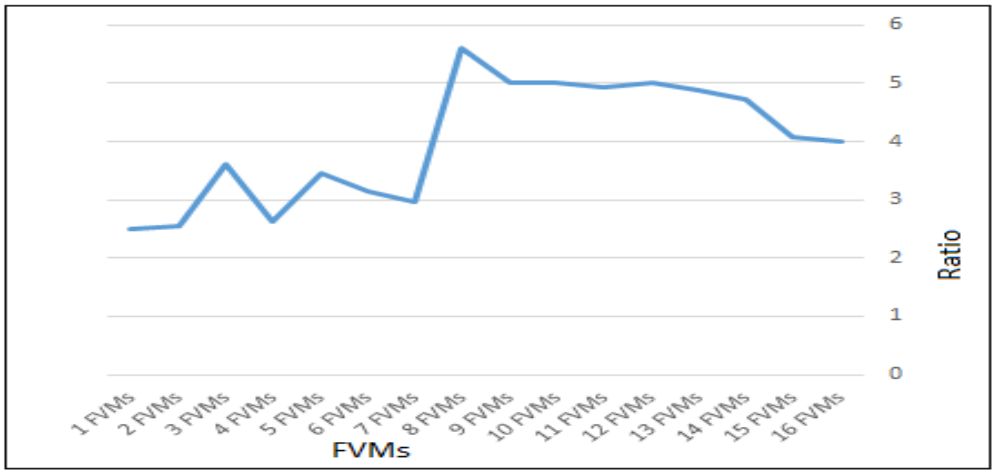


Figure 5.16: The ratio of the best result of Randomised local Reduction compared to Deterministic coordinated Reduction for Harmonic weights

max become stable after 7 FVMs for Deterministic coordinated Reduction (the mauve boxes). The second issue is that the highest range for the best result of Randomised-Local Reduction, the red boxes, is for 4 FVMs. No red boxes in the ranges bigger than 4. All of these suggest poor scalability in the case of harmonic series of weights. The final issue is related to the ratio between the best random reductions and the Deterministic-Coordinated one, which is generally bigger than the corresponding ratios for the previously considered sets of weights.

### 5.3.5 Exponential weights

Figure 5.17 for Exponential weights illustrates that a decrease in values of max weights in the first line of the mauve boxes row for Deterministic-Coordinated Reduction is for only 2 FVMs, and then it becomes stable for more FVMs. The value for more than two FVMs is 0.5. This result makes the Exponential weights the worst among the considered weights.

The red boxes illustrate that the best range for Randomised-Local Reduction, as shown in Figure 5.17, are in the range of 2 highest for all considered numbers of FVMs, which is another reason for claiming that the exponential set of weights is the worst to handle.

The ratio of the best value of Randomised-Local Reduction and the Deterministic Coordinated Reduction for exponential set of weights, as depicted in Figure 5.18, has a similar trend to the ratio for Harmonic weights. However, it reaches the peak earlier, for 2 FVMs, and then goes down faster with the growth of the number of FVMs. This is, however, not that optimistic in general, taking into account the previously mentioned lack of scalability - this trend

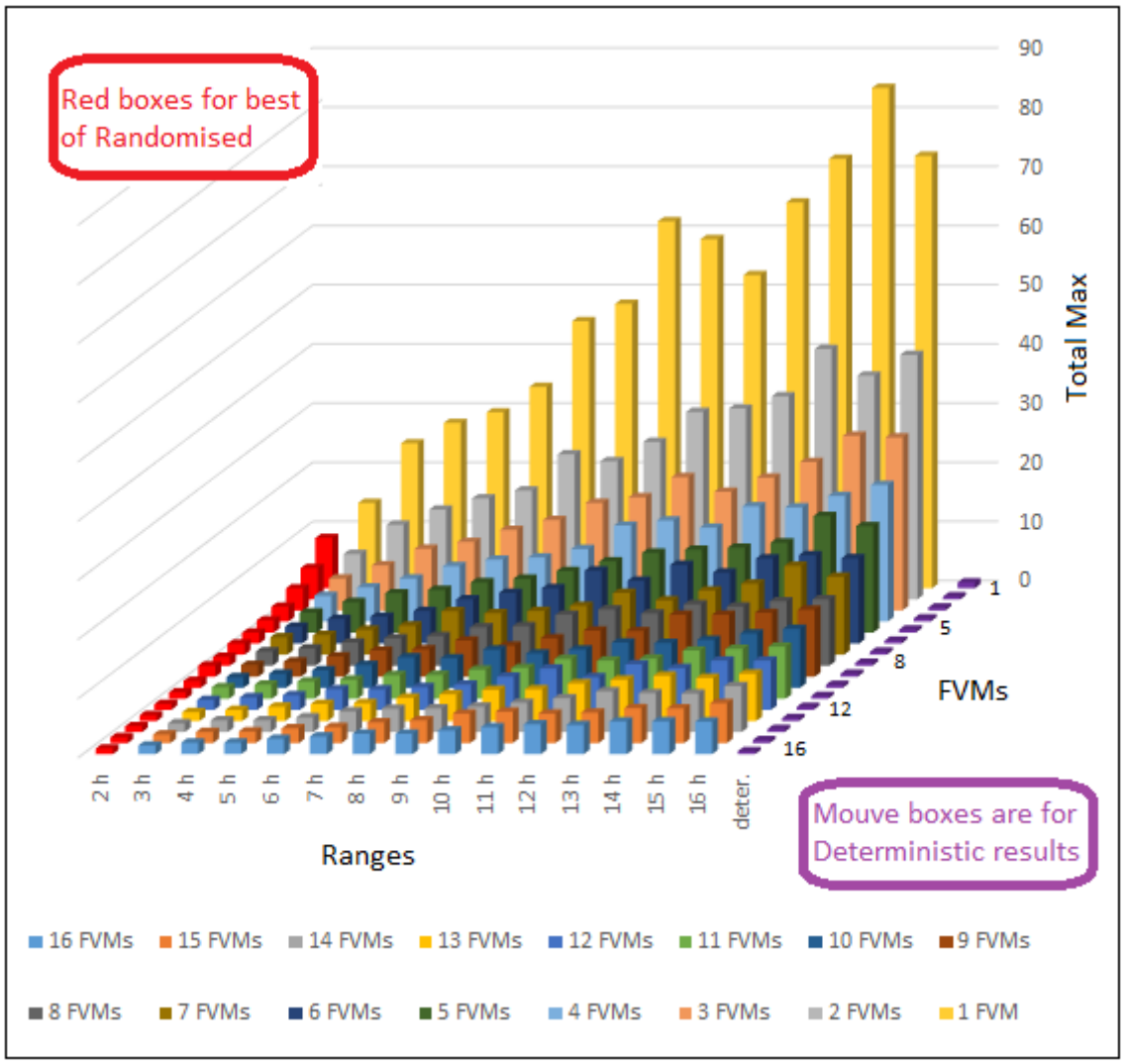


Figure 5.17: The total-max results for Deterministic-Coordinated version of Reduce-Max algorithm and for Randomised-Local version of Reduce-Max, taken for fifteen settings of the highest range, applied to Exponential weights

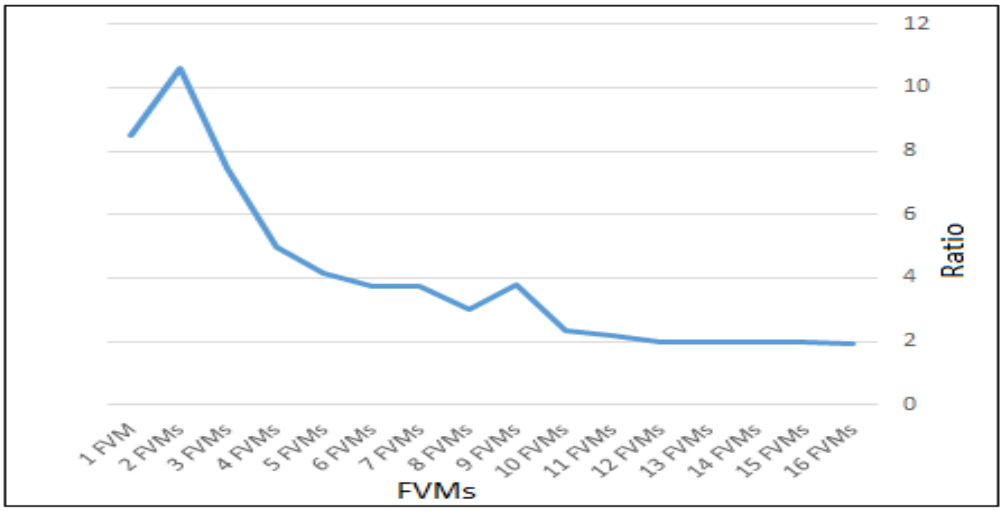


Figure 5.18: The ratio of the best result of Randomised-Local Reduction compared to Deterministic-Coordinated Reduction for Exponential weights

suggests only that both the Randomised-Local Reduction and the Deterministic-Coordinated Reduction are not scalable in a similar way for the Exponential set of weights.

### 5.3.6 Conclusion

Overall, the best results for Randomised-Local Reduction, which is the version of Reduce-Max requiring little resources and no coordination, occur in the case of Uniform weights. However, the results for Random weights and Arithmetic series of weights are close to those for the Uniform weights. However, The results for Harmonic and Exponential weights show worse results even in the case of Deterministic-Coordinated Reduction.

Deterministic-Coordinated results, which presented by mauve colour in all figures, are better than the results of Randomised Reduction. The ratio between them and the best of Randomised Reduction preformance depends on the use of data setting. However, Figures 5.27, 5.12 and 5.14 are similar but Arithmetic ratio less than others.

## 5.4 Symptoms-appearance results

In this section, we give and discuss the results of simulations for each set of input weights.

In the resulting graphs below, the fifteen rows of different colour boxes display maximum-weight of Reduce-Max in which FVMs choose VMs using the Randomised-Local reduction from the range of 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 and 16 of highest weight VMs, respectively. Sixteen columns correspond to different numbers of FVMs - from 1 to 16. The red colour boxes - one for each considered number of FVMs - denote the boxes with the smallest maximum weight among results for random selection from 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 and 16 highest latency VMs, i.e., indicate the best highest range (i.e., row id) to be used by Randomised Local reduction for each considered number of FVMs (column).

### 5.4.1 Random weights

Results in Figure 5.19 show that the more FVMs we apply for Randomised-Local Reduction, the wider range of highest values achieves the best total-max. This conclusion comes from the observation that the red box for one FVM is in the range of two highest values as the best for Randomised-Local Reduction. However, the red box moves towards the range of 15 highest values for 5 FVMs. It is also in the range of 16 highest values for only 7 FVMs. These results are close to the results for the same weights in Figure 5.9.

### 5.4.2 Uniform weights

Similarly to Random weights, the results of executing Reduce-Max algorithm for uniform weights in Figure 5.20 indicate that the first occurrence of the red box in the range of 16 highest is for 5 FVMs, while it is for 8 FVMs in 5.11. Also, it occurs here 12 times out of 16 in the range of 15 and 16 highest. However, the red box in these two ranges in 5.11 occur 11 times.

### 5.4.3 Arithmetic integer series weights

The results for Arithmetic integer sequences, cf., Figure 5.21, illustrate that the pattern of red boxes of choosing the best window of highest values for Randomised-Local Reduction is similar to the one in Figure 5.19 for the random set of weights. It is moving from the range of 2 highest through the range of 8 and 11 towards 16 highest for 1, 2, 3, and 6 towards 16 FVMs. However, the pattern of red boxes here is different to the one for Arithmetic weights

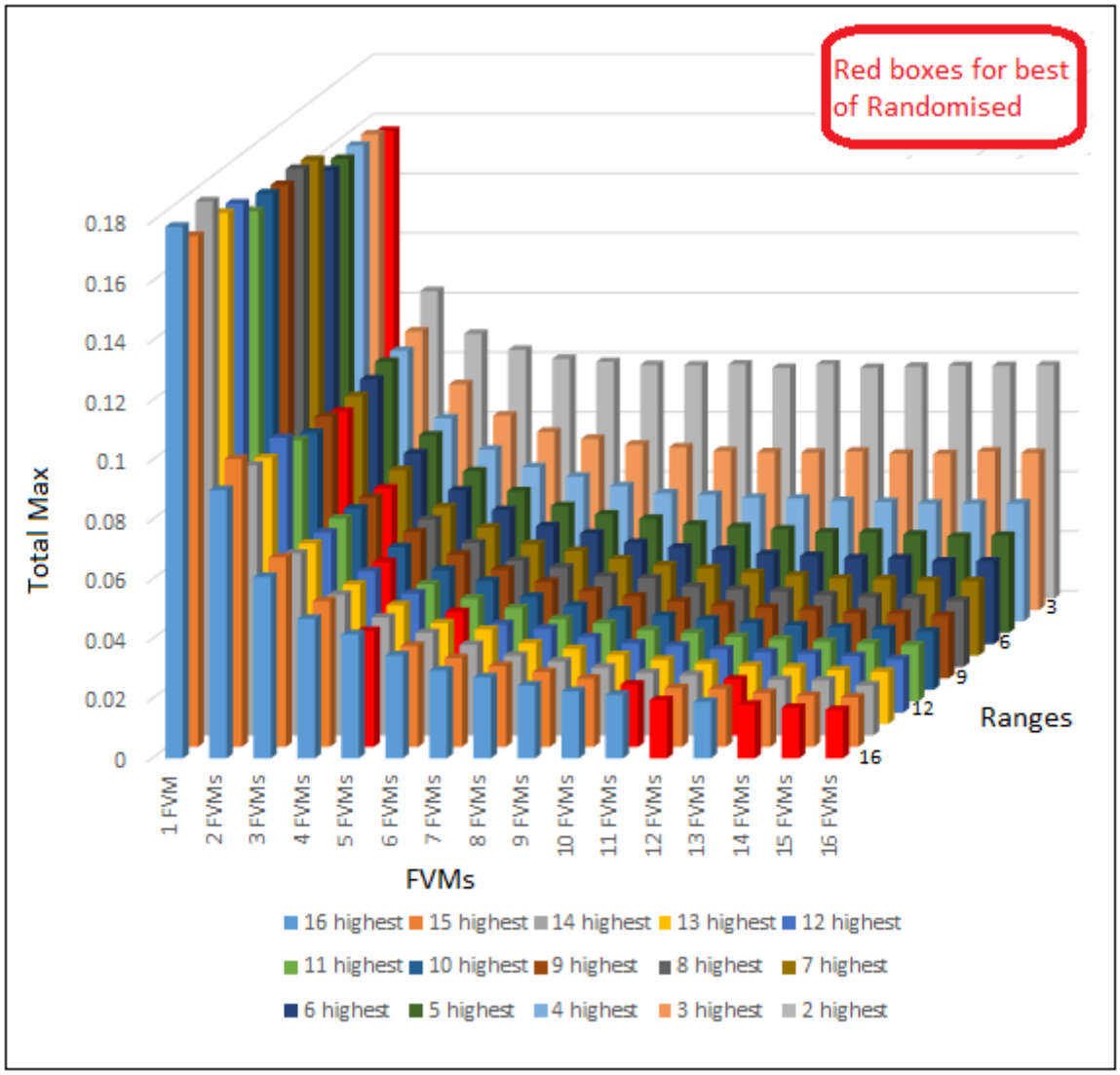


Figure 5.19: The total-max results for Randomised-Local version of Reduce-Max, taken after assumption of symptoms appearance for fifteen settings of the highest range, applied to Random weights.

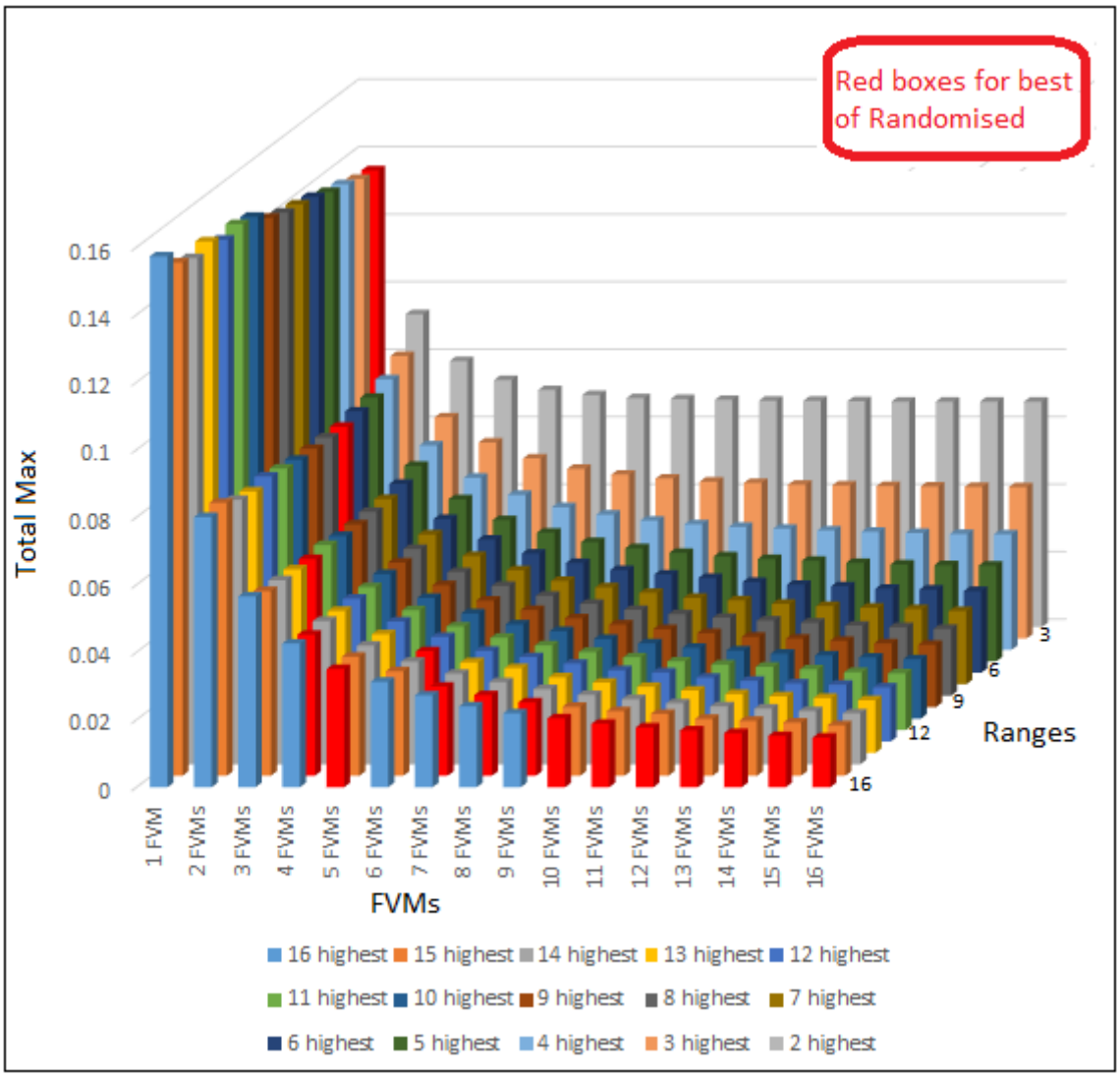


Figure 5.20: The total-max results for Randomised-Local version of Reduce-Max, taken after assumption of symptoms appearance for fifteen settings of the highest range, applied to Uniform weights.

without symptoms in Figure 5.13. The red box in it does not reach the range 16 until 13 FVMs are applied, whereas here in Figure 5.21, it reach 16 highest when only 6 FVMs are applied in the system. The result of reaching 16 highest give the system more scalability to choose from highest weighted VMs.

### 5.4.4 Harmonic weights

In the case of harmonic weights, the results in Figure 5.22 indicate that the red boxes of choosing best window of highest values for Randomised-Local Reduction are in the range of 2 highest through FVMs from 1 to 4. Then, it moves towards the range of 3 highest when 5 FVMs are applied and stays there also for 7 and 8 FVMs before it moves again to the range of 4 highest for 10 FVMs then towards 5 highest for 11 and 13 FVMs.

The pattern of red boxes here is close to the one for the same Harmonic set but without syptoms in Figure 5.15. However, the highest range that could be achieved in Figure 5.15 is 4 highest but the highest range here in Figure 5.22 is 5.

### 5.4.5 Exponential weights

Figure 5.23 for Exponential weights illustrates a similar pattern to harmonic sets with lack of respect to scalability to the number of FVMs. The red boxes illustrating the best range for random Reduction, as in Figure 5.23, are in the range of 2 highest for all considered numbers of FVMs except for 16 FVMs the best highest is in the range of 3 highest, which is the only difference with the result in 5.17 for exponential sets. This is the reason of claiming that the exponential set of weights is the worst to handle due to the lack of scalability.

### 5.4.6 Conclusions

In this section, an attempted was about to show that Reduce-max algorithm is behaving well with the appearance of symptoms. This leads to detect malicious behaviour in clouds early. The results here show that the execution of Reduce-Max algorithm is behaving well compared to the results present earlier for the two versions of Reduce-Max and that for Random, Uniform and Arithmetic weights. This conclusion is because the red boxes are achieving the highest ranges of highest VMs for those set of weights. However, there is a similarity to the results presented in sections 5.3.4 and 5.3.5 for Harmonic and Exponential sets.

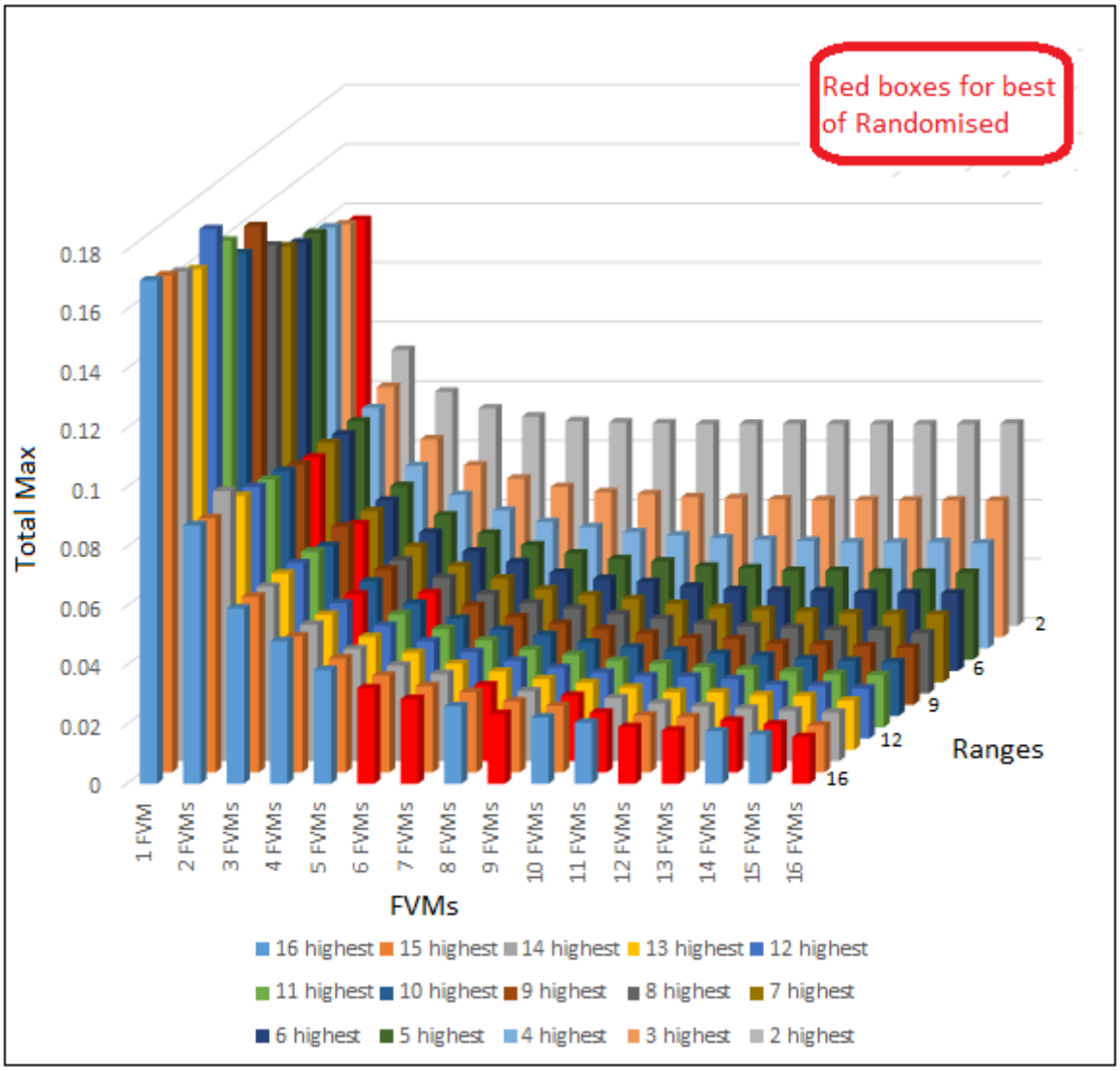


Figure 5.21: The total-max results for Randomised-Local version of Reduce-Max, taken after assumption of symptoms appearance for fifteen settings of the highest range, applied to Arithmetic weights.

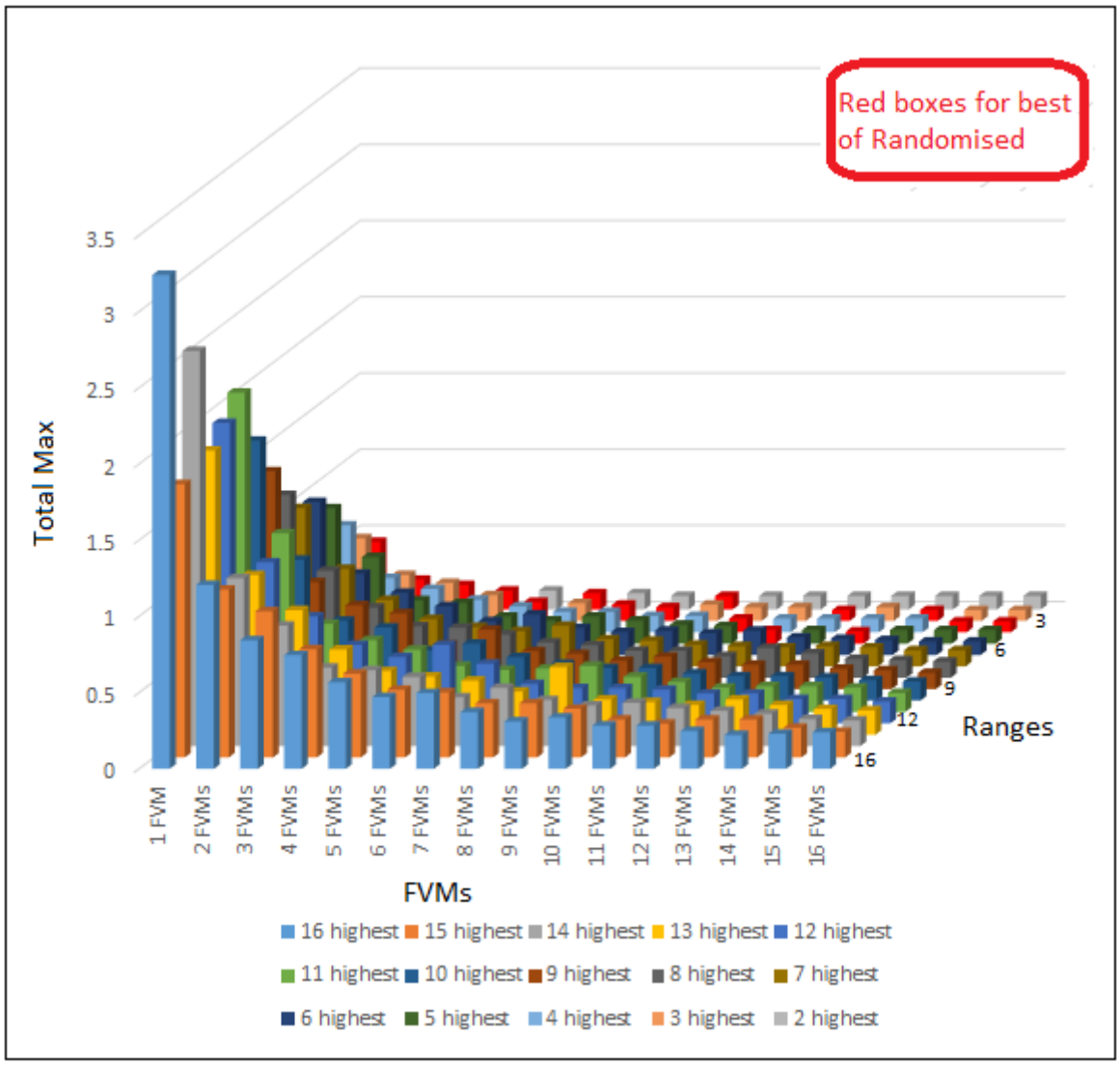


Figure 5.22: The total-max results for Randomised-Local version of Reduce-Max, taken after assumption of symptoms appearance for fifteen settings of the highest range, applied to Harmonic weights.

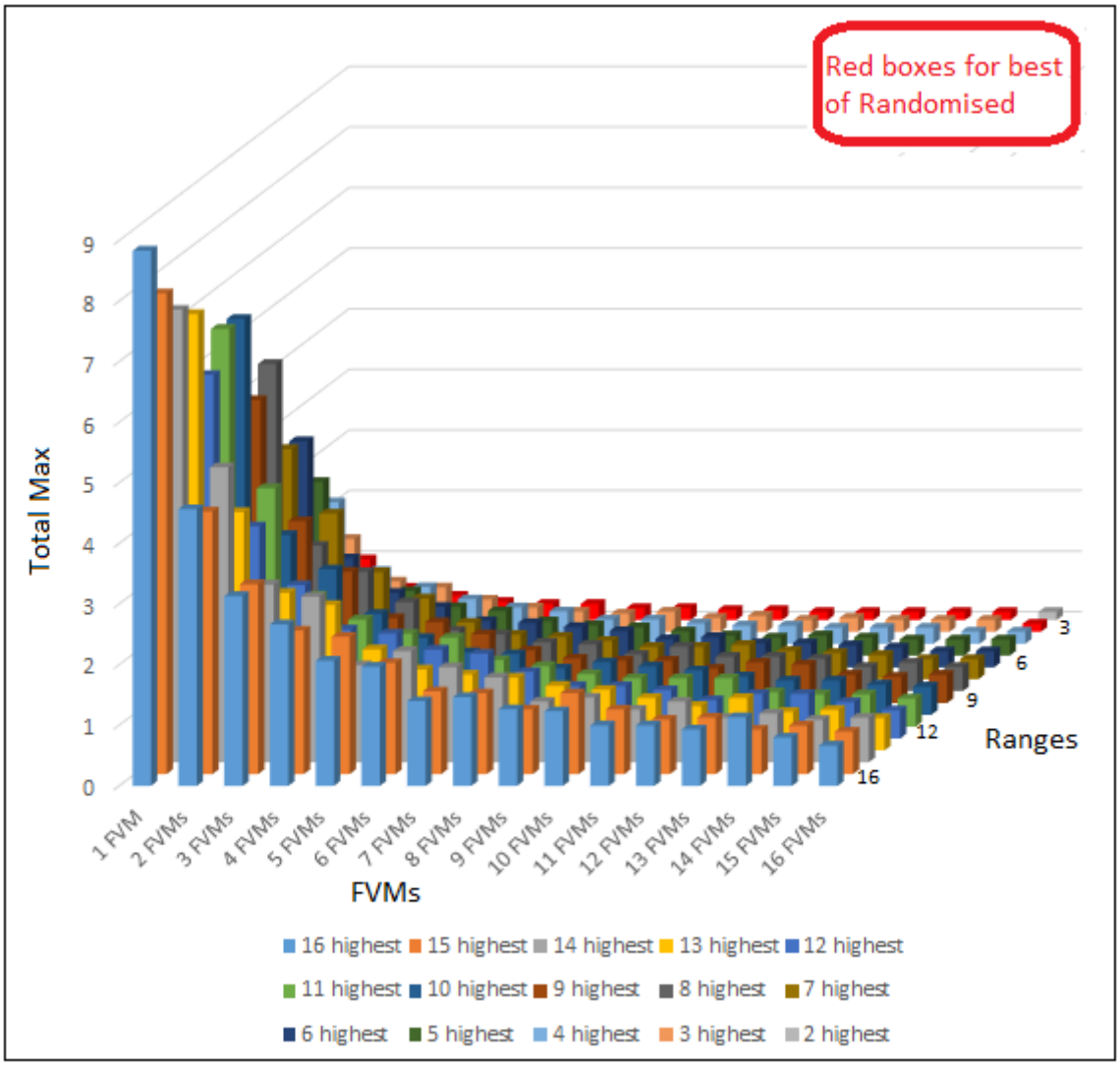


Figure 5.23: The total-max results for Randomised-Local version of Reduce-Max, taken after assumption of symptoms appearance for fifteen settings of the highest range, applied to Exponential weights.

## 5.5 Hierarchical structure of Reduce-Max algorithm

The results will be explained separately for the two questions that are posed for this model in previous chapters. It will be done that for each considered set of weights: Uniform, Random uniform integers of both ranges [1,3] and [1,15], and Poisson for both $\mu = 2,8$, and that for 16 FVMs. The first part of each subsection addresses the question how the number of clusters affects monitoring of VMs. The second part studies what the best ranges for random selection of next VM are for different numbers of clusters in the system and thus how parameter $\alpha$ influences the performance.

### 5.5.1 Uniform weights

Recall that Uniform weights means that all VMs are equal in regards to the set of weights. Next the results will be presenting firstly in regards to the performance of the number of clusters. The second subsection is about The performance regarding the value of $\alpha$ that makes range of highest VMs wider.

#### The performance regarding the number of clusters

From Figure 5.24, it can be sees clearly that the results of highest total-max for Deterministic-coordinated algorithm displayed by “the red boxes”, look similar across all clusters. However, for Randomised Local Reduction the highest total-max is getting better results when a bigger number of clusters applies. The best total-max of applying 16 FVMs across all values of $\alpha$ is 0.132 and is achieved for 16 clusters in the system. Moreover, the results of Deterministic-coordinated Reduction is 0.062 for all number of clusters. This is because the load for FVMs is the same for all number of clusters. For 1 cluster, 16 FVMs Vs 1024 VMs, whereas there is 1 FVM Vs 64 VMs for 16 clusters.

In brief, the behaviour of stripped boxes as best results of Randomised-Local Reduction is getting closer to red boxes of Deterministic-coordinated results with a bigger number of clusters. These results lead to the conclusion that bigger number of clusters applied in the system is better for Randomised-Local Reduction. A possible explanation of this claim is that the ranges of highest are smaller with a larger number of clusters. Then the possibility of hitting the highest is higher.

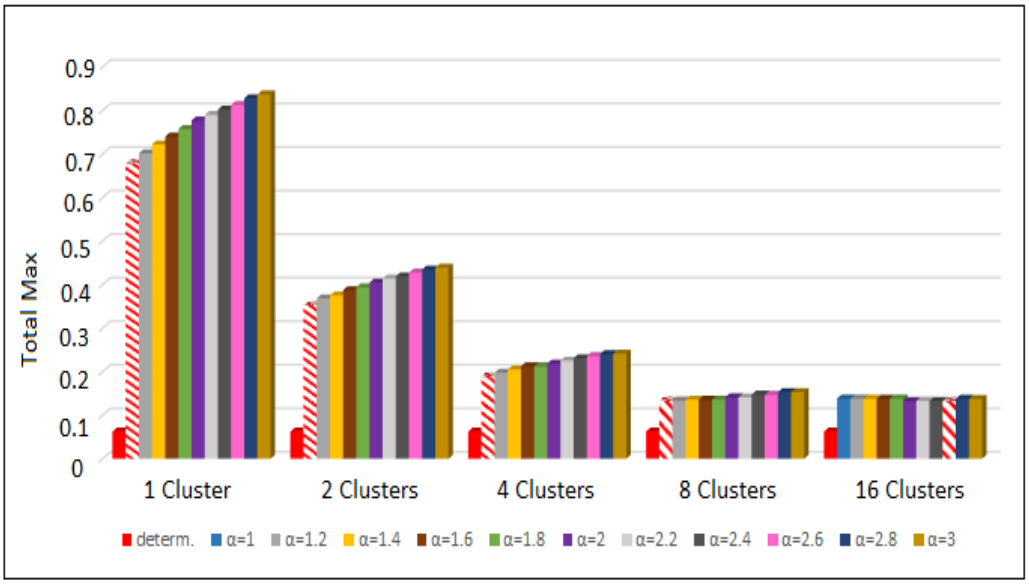


Figure 5.24: Total-max for Uniform weights for different numbers of clusters

## The performance regarding the value of $\alpha$

In Figure 5.24 prepared for Uniform weights and 16 FVMs, choosing randomly from different ranges of highest VMs according to the value of $\alpha$, the best results (in marked columns with red-white strips) among all considered values of $\alpha$ for every considered number of clusters occur four times when $\alpha = 1$. however, the best result of marked columns with red-white strips occur once when $\alpha = 2.6$ for 16 clusters. The best total-max, over all values of $\alpha$ and clusters' numbers, is about 0.132 and is achieved for 16 cluster and $\alpha = 2.6$. Therefore there is a clear choice of best value of $\alpha$, and this is for best settings of cluster numbers the best value of $\alpha$ seems to be closer to 1. For the same number of clusters the best total-max is getting better with smaller value of $\alpha$ for the number of clusters 1, 2, 4 and 8. A possible explanation for this claim is that the ranges of highest are smaller with smaller values of $\alpha$.

Regarding the ratio between the best results of Randomised-Local and the optimal results of Deterministic-Coordinated in Figure 5.25, it starts from the highest and worst point around 11 when one cluster applied. Then it dropped dramatically to be under 6 for 2 clusters. Finally, it reaches the best point of 2 for 8 and 16 clusters.

Overall, the results for 16 FVMs in case of uniform weights seem interesting because of the following reasons. First, the results for 16 clusters are mostly better for all values of $\alpha$ than other weights. This means that hierarchical structure and clustering works well for the approached system. Second reason is related to the best marked column. It is mostly when

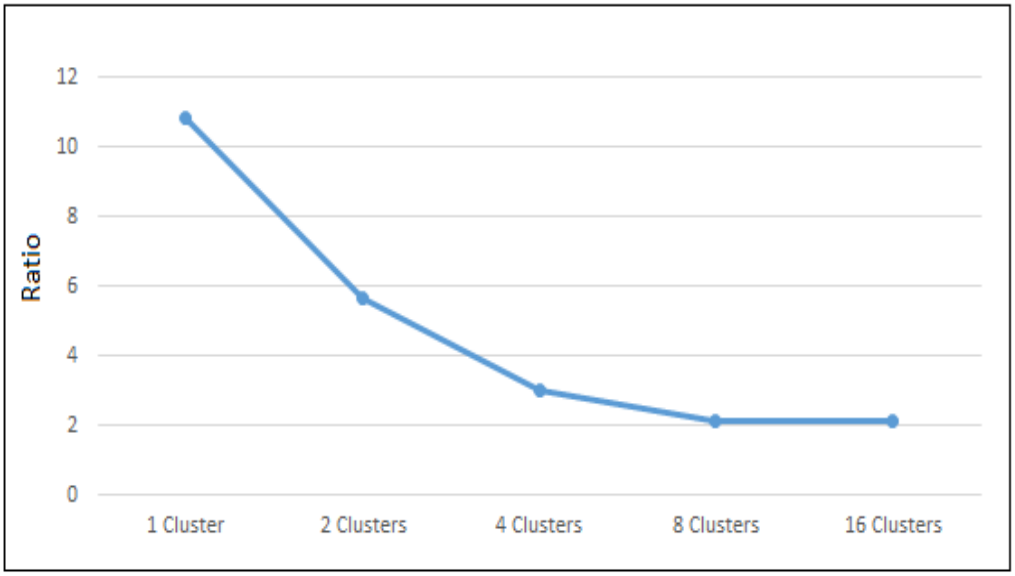


Figure 5.25: The ratio between best results of Randomised-Local Reduction and results of Deterministic-Coordinated Reduction for Uniform weights

$\alpha = 1$. This suggests that the number of FVMs in the system can work more in an efficient way with smaller ranges. Finally, the best total-max for marked columns are getting better and not far from the optimal results of Deterministic-Coordinated Reduction that are in red columns in the figure, when a larger number of clusters is applied.

## 5.5.2 Random weights

Random weights mean that the values of weights are integers selected randomly and independently from the range of integer values. In our case, there are two ranges of selected number (as a weight of a VM). The first one is between 1 and 3 and the second is between 1 and 15, see also [3].

### 5.5.2.1 Random [1,3] weights

Here we explain the results based on that the selection of weights is taking randomly from the range between 1 and 3. The results in Figure 5.26 show that the marked columns for all considered numbers of clusters occur when $\alpha = 1$. The reason behind this behaviour is that the ranges of highest is smaller with smaller value of $\alpha$.

The best (i.e., smallest) value of total-max, over all values of $\alpha$ and all considered numbers of clusters, is about 0.14 for 8 clusters. This number is bigger than the corresponding one

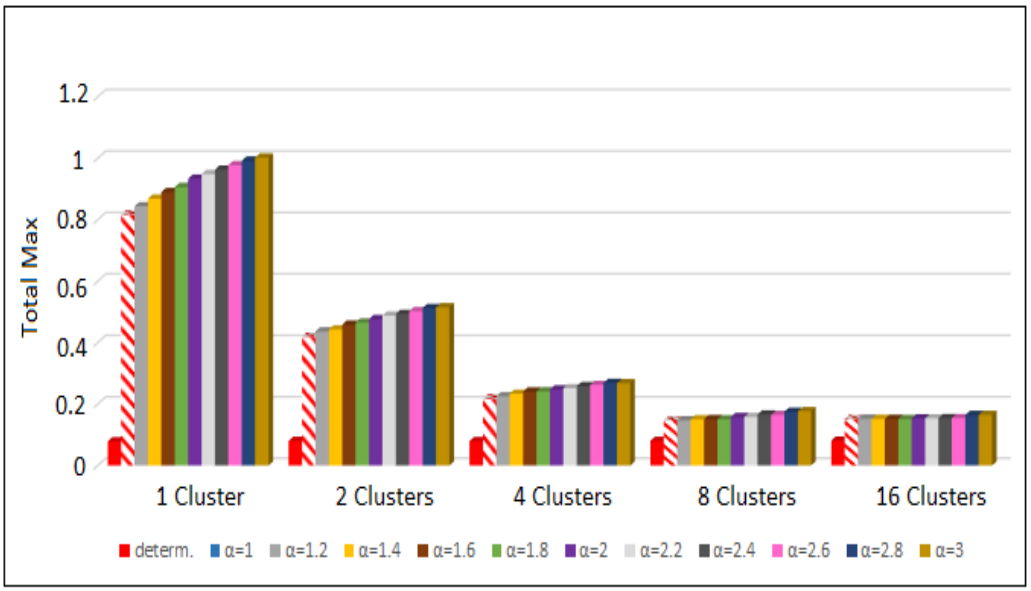


Figure 5.26: Total-max for Random [1,3] weights for different numbers of clusters

for the same number of FVMs in the case of Uniform weights. The worst value of total-max is around 1 and it happened with one cluster applied. The optimal results here of Deterministic Reduction started from 0.078 for one cluster and ended with 0.081 for 16 clusters. These results are also worse than 0.062 for Uniform weights.

### The performance regarding the number of clusters

The total-max behaviour changes with the number of clusters. In Figure 5.26, we see mainly that the results of total-max are getting better across all values of $\alpha$ with the increasing number of clusters. The best result occurs with 8 clusters, which is around 0.14. However, for 1 cluster the total-max could reach as bad as 1.003, that is the worst result for random weights when 16 FVMs are applied. The optimal results vary over the number of clusters, from 0.078 for 1 cluster to 0.081 for 16 clusters.

### The performance regarding the value of $\alpha$

For this aspect, in Figure 5.26, the marked columns are for $\alpha = 1$ for all number of clusters are implemented. Therefore there is a clear choice of best value of $\alpha$, and this is for best settings of cluster numbers the best value of $\alpha$ seems to be almost 1. For the same number cluster the

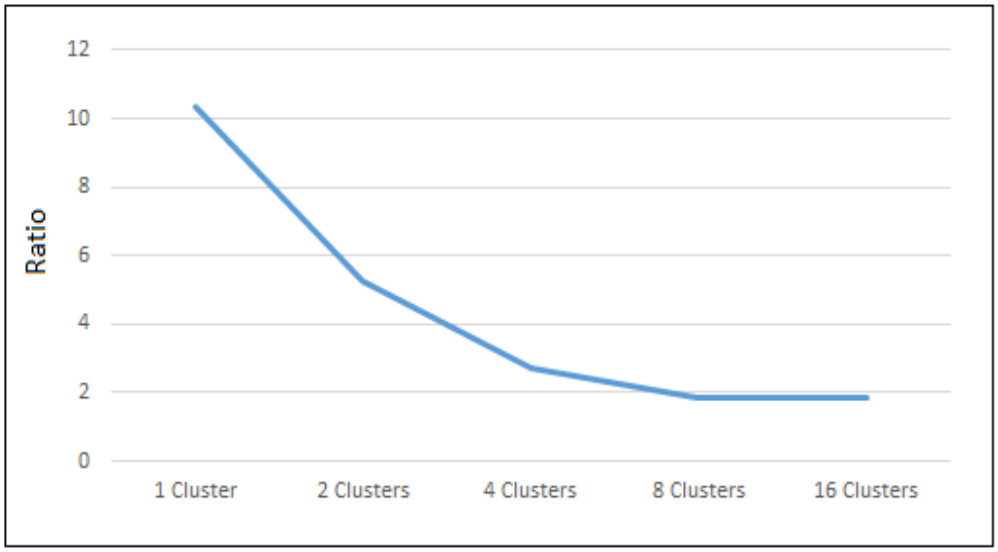


Figure 5.27: The ratio between best results of Randomised-Local Reduction and results of Deterministic-Coordinated Reduction for Random [1,3] weights

total-max is getting better with smaller value of $\alpha$ for all considered numbers of clusters. The reasons of this claim is that the ranges of highest are getting smaller with smaller value of $\alpha$.

Regarding the ratio between the best result of Randomised-Local Reduction and the results of Deterministic-Coordinated in Figure 5.27, it starts from worst point around 10 when one cluster is applied. Then it dropped dramatically to be around 5 for 2 clusters. Finally, it reaches the best point of 2 for 8 and 16 clusters.

Overall, the results for Random weights with 16 FVMs are worse than the analogous for Uniform weights. The reasons are that, firstly, the best total-max of the value of 0.14 for Random weights is worse than it was in the case of Uniform weight (the value was 0.13). Second reason is related to the difference between them for the optimal Deterministic Reduction. It is 0.062 for Uniform, whereas, it is around 0.08 for Random weights. Finally, the worst values of total-max are also worse in the case of Random weights. They are 1.003 versus around 0.83 for Uniform weights.

#### 5.5.2.2 Random [1,15] weights

For 16 FVMs, each choosing randomly from different ranges of highest VMs according to the value of $\alpha$ for random weights selected from the range [1,15], it appears that the best total-max for all $\alpha$ across all number of clusters is about 0.15, cf., Figure 5.28. It is again worse

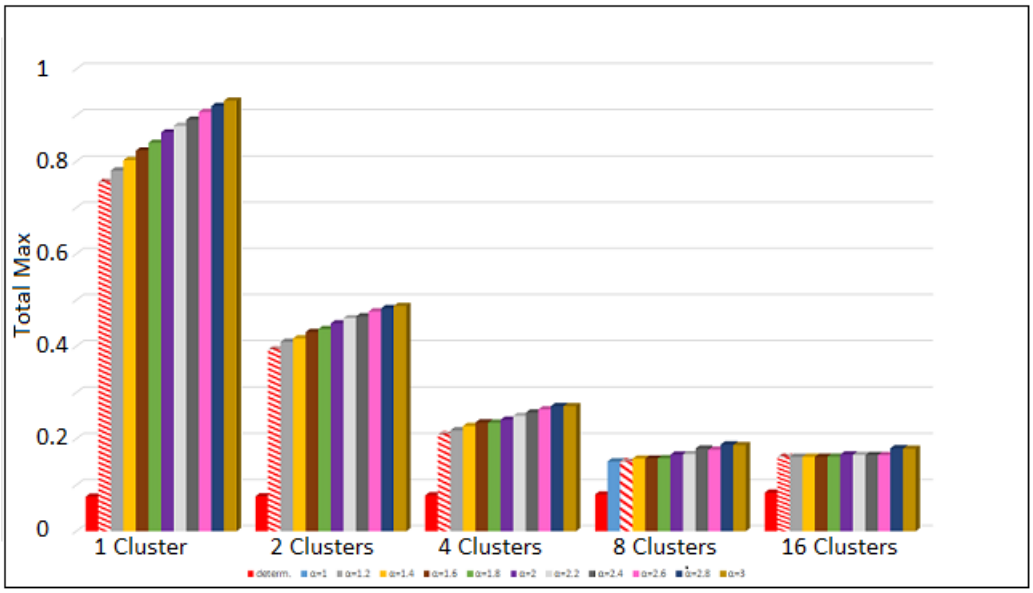


Figure 5.28: Total-max for Random [1,15] weights for different numbers of clusters

than it was with Uniform weights for the same number of FVMs (value 0.13) and Random of 3 (value 0.14). However, the worst total-max here 0.93 that is better than the worst total-max for random of the range [1,3] (the value was 1.003).

## The performance regarding the number of clusters

For the varying number of clusters in Figure 5.28, the behaviour of the total-max performance is not much different from the setting with random of the range [1,3]. The results of total-max for all consider numbers of clusters decrease with the increasing number of clusters. Moreover, the total-max values for all numbers of clusters again are behaving well with more clusters. The results of applying one cluster across all values of $\alpha$ is around 0.85, but for 16 clusters it is approximately around 0.16.

Deterministic results here start from the total-max of 0.074 for one cluster to end with 0.83. The results are different from Random [1,3] integers and of course from Uniform weights. These results lead to confirm that random weights are worse than the uniform ones but not that far.

## The performance regarding the value of $\alpha$

Regarding the efficiency for different values of parameter $\alpha$, in Figure 5.28, the best total-max among all considered numbers of clusters is in the case of 1 cluster and the marked column of

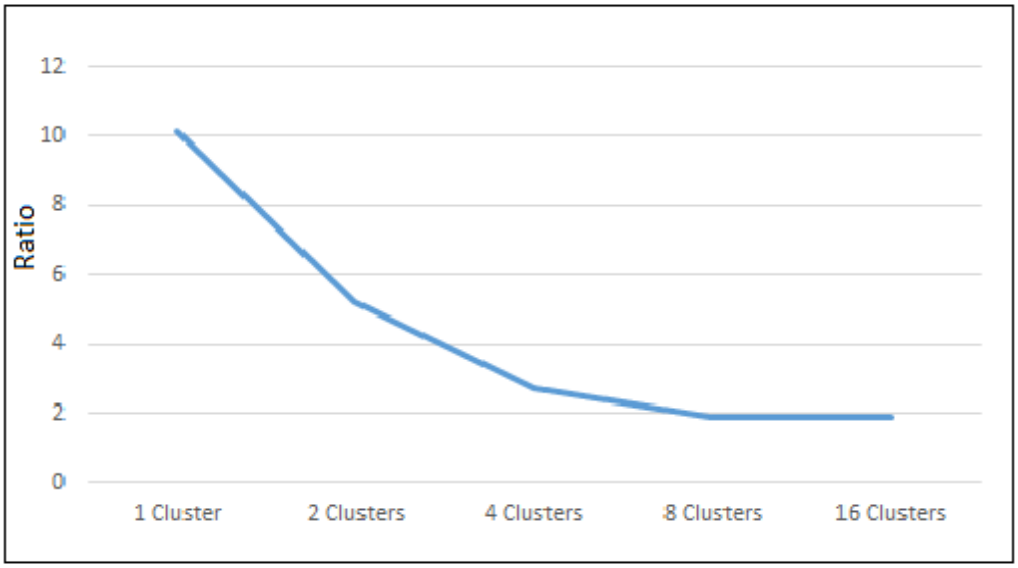


Figure 5.29: The ratio between best results of Randomised-Local Reduction and results of Deterministic-Coordinated Reduction for Random [1,15] weights

$\alpha = 1$. Apart of the marked column for $\alpha = 1.2$ when there are 8 clusters, all other marked columns are for $\alpha = 1$.

The difference for the ratio between best result of Randomised-Local Reduction and the optimal results of Deterministic-coordinated Reduction, cf., in Figure 5.29 from other ratios is that it drops quicker. It starts from the worst point around 10 when one cluster is applied. Then it drops dramatically to be around 5 for 2 clusters. Finally, it reaches the best point of 2 for 8 and 16 clusters.

Overall, the following two observations could be derived from the results for 16 FVMs for Random weights of 15. First, the results for 8 and 16 clusters are better, for almost all values of $\alpha$, than those for smaller numbers of clusters. Second, Random weights give worse results in comparison with Uniform weights but the difference is not critical.

### 5.5.3 Poisson weights

Poisson distribution generates values for VMs weights according to the value of the parameter $\mu$, where $\mu$ is a given value either 2 or 8 in the experiments. We will explain the results for both $\mu$ equals 2 or 8.

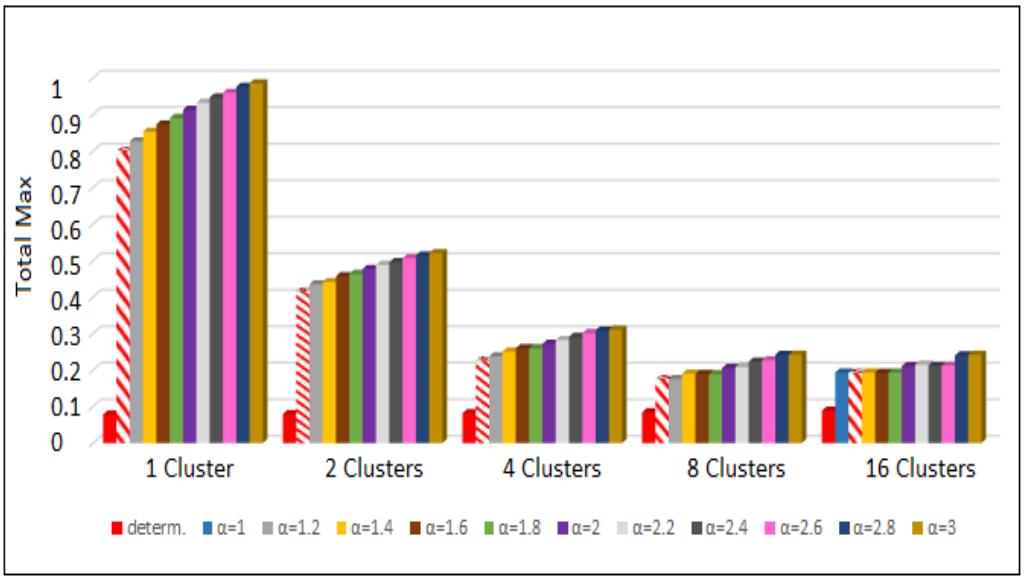


Figure 5.30: Total-max for Poisson $\mu$ 2 for different numbers of clusters

#### 5.5.3.1 Poisson (2) weights

As we can see in Figure 5.30, the best total-max achieved for Randomised-Local Reduction over all values of $\alpha$ and across all numbers of clusters is about 0.17. That happens for 8 clusters in the system. To compare, this best result of Poisson when $\mu$ equals 2 is worse than to the best one of Uniform weights and both Random weights.

## The performance regarding the number of clusters

The behaviour of total-max in regards to the number of clusters in Figure 5.30 is similar to the corresponding picture for both Random and Uniform weights. The results of total-max for Randomised-Local Reduction across all numbers of clusters are getting better across all values of $\alpha$ with the increasing number of clusters. However, the results of applying 16 FVMs for 8 clusters, across smaller values of $\alpha$, are better than the same values of $\alpha$ for 16 clusters.

To have some comparisons, the results of optimal Deterministic-Coordinated Reduction start from 0.078 for one cluster and end with 0.088 for 16 clusters. This value of 0.088 is the worst value for Deterministic-Coordinated Reduction among all studied data sets.

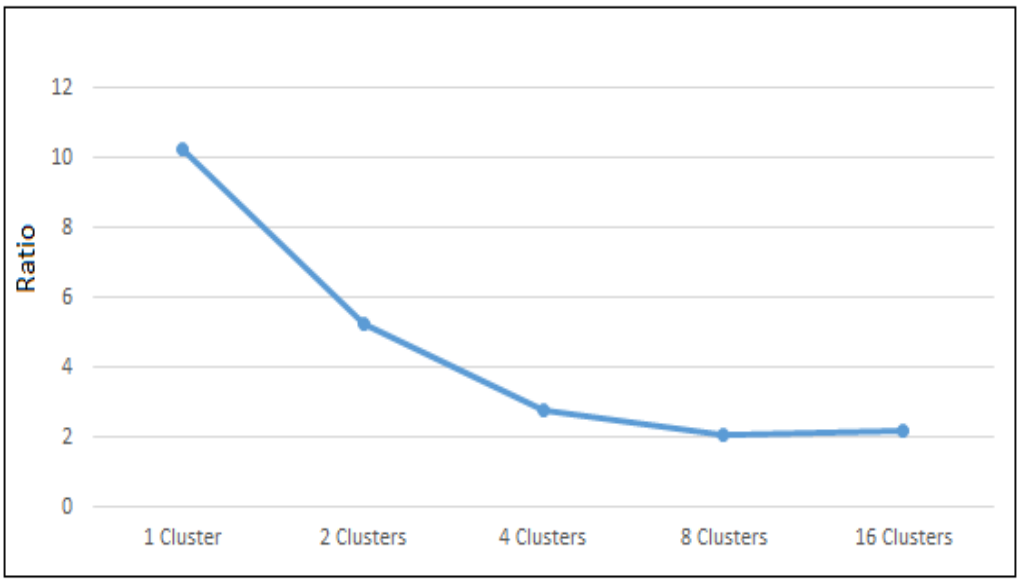


Figure 5.31: The ratio between best results of Randomised-Local Reduction and results of Deterministic-Coordinated Reduction for Poisson $\mu$ 2

## The performance regarding the value of $\alpha$

Related to the impact of parameter $\alpha$, it could be seen in Figure 5.30 that the total-max performance is not much different from the ones for previous weights. It is also almost similar from perspective of positions of the marked columns. There are four marked columns when $\alpha = 1$ for all number of clusters. However, the last marked column for 16 clusters is when $\alpha = 1.2$. The best total-max among all numbers of clusters — in the case of 8 clusters — is when $\alpha = 1$.

The ratio between best results of Randomised-Local Reduction and the optimal results of Deterministic-Coordinated Reduction, as in Figure 5.31, starts from the point around 10 when one cluster is applied. Then it drops to be just under 6 for 2 clusters. After that, it is about 3 for 4 clusters. Finally, it reaches the best point of 2 for 8 clusters then goes just above 2 for 16 clusters.

Overall, the results for Poisson weights with $\mu = 2$ are also interesting. It is true that it is not far from the previously studied weights, but it has the worst results for optimal Deterministic-Coordinated Reduction. Not only that its best result of Randomised-Local Reduction is worse than the best one of the previous weights.

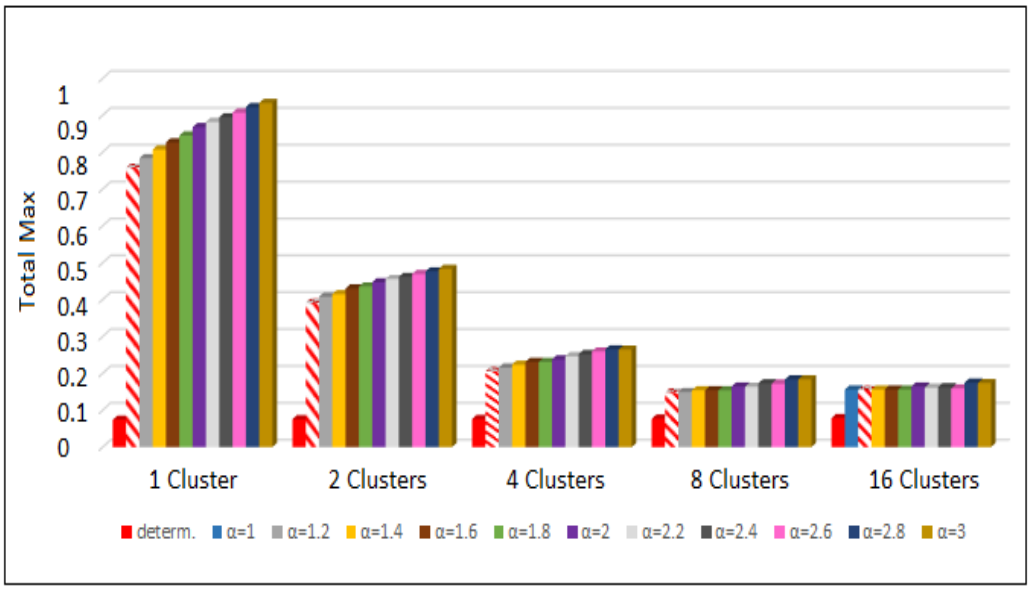


Figure 5.32: Total-max for Poisson weights $\mu$ 8 for different numbers of clusters

#### 5.5.3.2 Poisson (8) weights

In Figure 5.32, with 16 FVMs choosing randomly from different ranges of highest VMs for Poisson weights with $\mu = 8$, the best total-max for all $\alpha$ is about 0.014. This result is really close to both random weights (value of 0.014) for random of[1,3] and (value of 0.015) for random of [1,15]. In addition, this result is better than the best result of Poisson weights when $\mu = 2$ (value of 0.017).

## The performance regarding the number of clusters

Regarding the number of clusters, in Figure 5.32, it can be seen that the results of total-max for Randomised-Local Reduction are getting better with increasing number of clusters across all values of $\alpha$. However, the results for 8 and 16 clusters are close to each other. The results of applying 16 FVMs for 2 clusters across all values of $\alpha$ are around 0.43, and for 4 clusters they are about 0.23. Finally, for 8 and 16 clusters they are mostly around 0.16.

The results of red columns for optimal Deterministic-Coordinated Reduction start from 0.073 for one cluster and end with 0.078 for 16 clusters. These results of Deterministic-Coordinated Reduction for Poisson weights with $\mu = 8$ are considered the second best results after the one for Uniform weights.

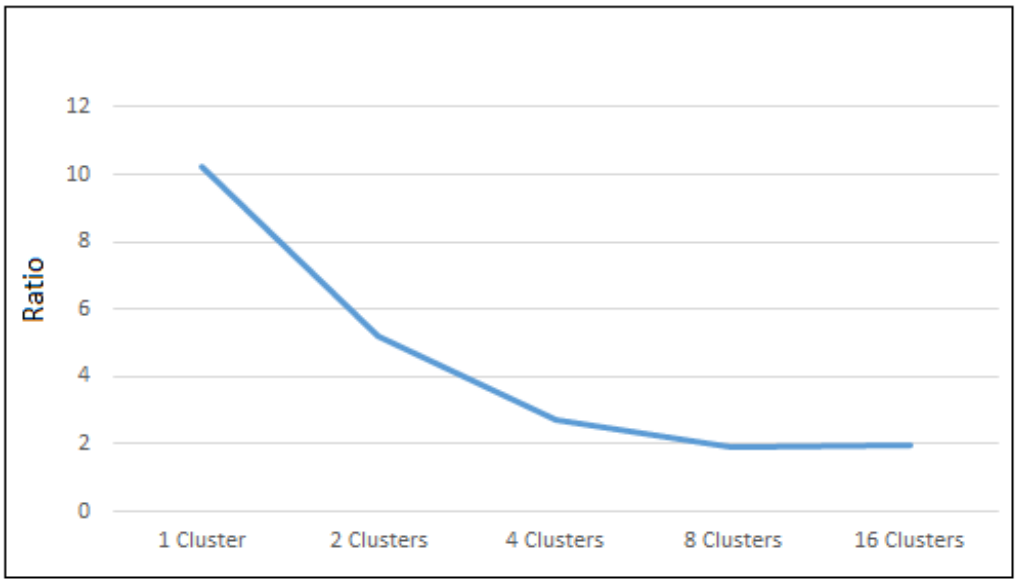


Figure 5.33: The ratio between best results of Randomised-Local Reduction and results of Deterministic-Coordinated Reduction for Poisson $\mu$ 8

### The performance regarding the value of $\alpha$

Regarding the impact of $\alpha$, the interesting thing is that the marked column exists in same positions like in the case of Poisson for $\mu = 2$. It exists four times when $\alpha = 1$, and one time for $\alpha = 1.2$, and no marked column occurs for other alpha. The best total-max among all numbers of clusters, achieved in the case of 8 clusters, is when $\alpha = 1$.

The ratio between best results of Randomised-Local Reduction and optimal results of Deterministic-Coordinated Reduction in Figure 5.33, starts from the point just above 10 when one cluster is applied. Then it drops to be just around 5 for 2 clusters. After that, it reaches the best point of 1.9 for 8 clusters and then goes just under 2 for 16 clusters.

Overall, there are two positive points in case of Poisson weights with $\mu = 8$: this setting provides the second best results after the ones with Uniform weights, and the best performance is reached for the smallest range of parameter $\alpha$.

## 5.5.4 Conclusions

In this section, we used a hierarchical structure of cloud to study efficiency of monitoring VMs in order to discover symptoms. The structure includes three levels: virtual gateway, virtual

clusters and virtual nodes. The simulations were done to examine the impact of hierarchical structure from the perspective of cloud monitoring. They were done to compare the results for different numbers of clusters, including one cluster that corresponds to previously considered homogenous systems, under the same number of FVMs. We used three popular sets of data to model weights of VMs: Uniform, Random for two ranges of weights (selection from 1 to 3 and the other one is a selection from 1 to 15) and Poisson for two $\mu = 2$ and $\mu = 8$, and considered different ranges of highest range for random selection of next VM according to the value of $\alpha$.

From simulations' results, it is recommended to use hierarchical structure because it is effective with all the considered sets of weights. In all sets of weights and different numbers of FVMs, the results are better when the number of applied clusters is increasing, although, generally, the result for 8 is similar to 16 clusters. This, together with 1024 VMs in total, suggests 128 VMs in a cluster when using 16 FVMs in the whole system.

In regards to the sets of weights, Uniform weights then Poisson for $\mu = 8$ are better tractable than the other considered weights, although the differences are not critical. For the question of best choice of highest ranges to implement random Reduction, we get in general better results for smaller values of $\alpha$, generally when $\alpha = 1$. What is most important, the trends in performance depend primarily on the number of clusters in all the considered settings. This confirms how important is proper structuring of the cloud for efficiency of the monitoring process, in particular, for implementing light security and resiliency systems on the top of the cloud.

## 5.6 Random-Start-Round-Robin algorithm

In this section, the results of simulations for five different data sets of input weights are presented and discussed for Random-Start-Round-Robin algorithm.

### 5.6.1 Uniform weights

We start with the results of Uniform weights studied for Random-Start-Round-Robin algorithm. The experiments were done using 16 FVMs and for 1, 2, 4, 8 and 16 Clusters as following.

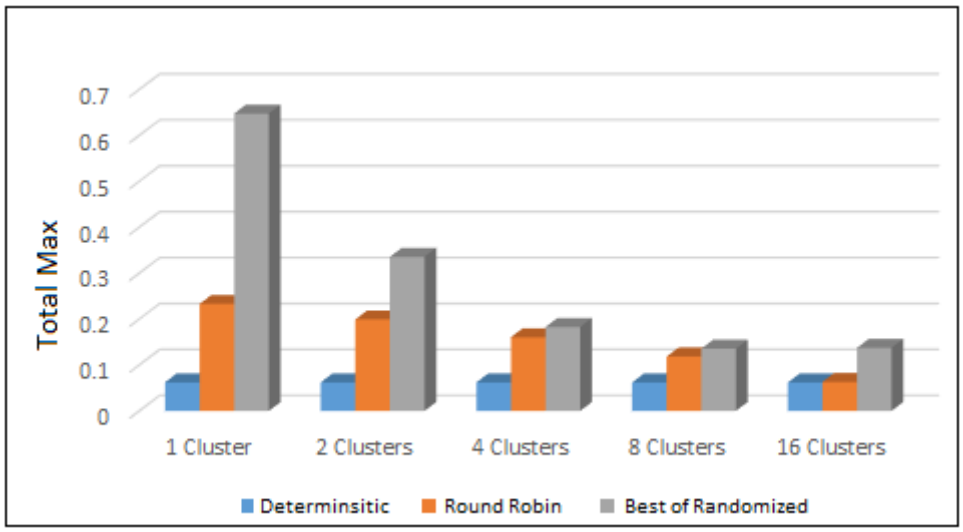


Figure 5.34: Total-max of Random-Start-Round-Robin algorithm for Uniform weights

In Figure 5.34, it can be seen clearly that the results of total-max of Random-Start-Round-Robin algorithm are much better of the best result of Randomised-Local version of Reduce-Max algorithm, especially for smaller numbers of clusters.

The result in Figure 5.34, for one cluster in the system is around 0.23, which is better than the best result of Randomised-Local version that is above 0.6. However, the Deterministic-Coordinated version of Reduce-Max with the same settings shows the result of 0.062 that is not only better than the Randomised-Local version but also better than Random-Start-Round-Robin algorithm's result.

Then, the results of Random-Start-Round-Robin go down towards 0.063 when 16 clusters applied, which is really close to Deterministic-Coordinated results. Moreover, the best results of Randomised-Local version of Reduce-Max are decreasing too with the increasing number of clusters. The best results of Randomised-Local Reduction, when 16 clusters are in the system, is around 0.13.

### 5.6.2 Random [1,3] weights

The results for Random [1,3] weights, when the selection is from the range from 1 to 3, in Figure 5.35 are better than the best results of Randomised-Local version of Reduce-Max algorithm for most of the numbers of clusters, and worse than Deterministic-Coordinated results in all cases.

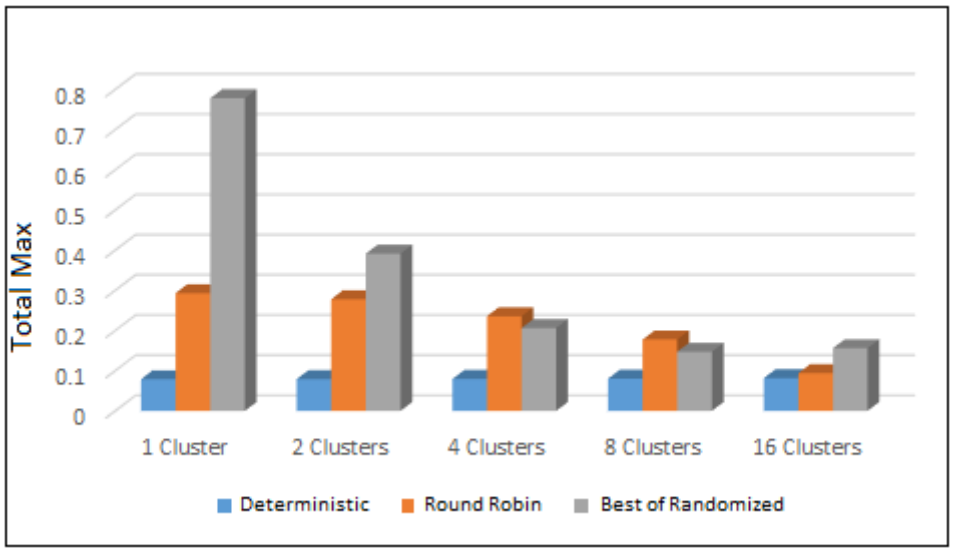


Figure 5.35: Total-max of Random-Start-Round-Robin algorithm for Random [1,3] weights

The results start from around 0.29 for one cluster and end with 0.094 for 16 cluster. The case with the best result of Randomised-Local version of Reduce-Max is almost the same. It starts from about 0.78 for one cluster and ends with around 0.16 for 16 clusters. However, the results of Deterministic-Coordinated version increase from 0.079 for one cluster to 0.081 for 16 clusters.

### 5.6.3 Random [1,15] weights

In this subsection, results are shown for random set when the selection is from 1 to 15 for different numbers of clusters.

Overall, in Figure 5.36 the results of Random-Start-Round-Robin are similar to those for random [1,3] weights. They are better than the best results of Randomised-Local version of Reduce-Max algorithm for most of the numbers of clusters, and worse than Deterministic-Coordinated results in all cases. They start from about 0.43. This value is close to the corresponding one for Poisson (8) weights. This value become 0.12 when 16 FVMs are applied.

The result of Deterministic-Coordinated version is about 0.074 for one cluster and goes up till 0.08 for 16 clusters. Moreover, the best results of Randomised-Local version decrease from 0.72 to 0.16.

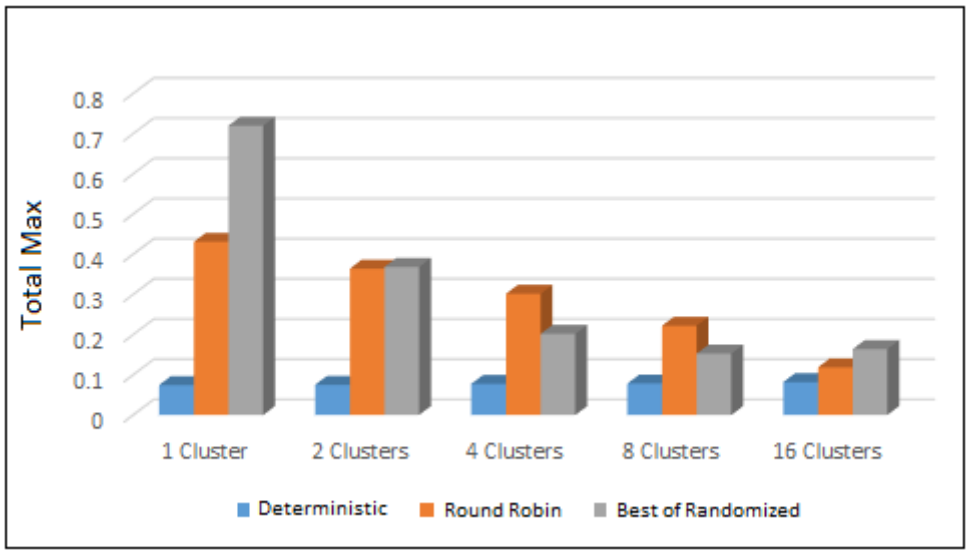


Figure 5.36: Total-max of Random-Start-Round-Robin algorithm for Random [1,15] weights

### 5.6.4 Poisson (2) weights

For Poisson weights with $\mu = 2$, in Figure 5.37, the result of Random-Start-Round-Robin algorithm is showing the worst results compared to the results of both Deterministic and Randomised versions of Reduce-Max.

Random-Start-Round-Robin result for one cluster is approximately 0.875. This value of total-max is decreasing with the rising number of clusters. It reaches around 0.26 when 16 clusters are in the system. The same case happens with the best result of Randomised-Local version of Reduce-Max. The total-max starts from about 0.77 for one cluster and ends with around 0.21 for 16 clusters.

On the other hand, the results of Deterministic-Coordinated version of Reduce-Max start from approximately 0.077. Then, an increase to the results happens until reaching 0.088 for 16 clusters. However, the results for Deterministic-Coordinated are indicating the best total-max.

### 5.6.5 Poisson (8) weights

In regards to Poisson weights with $\mu = 8$, the results of Random-Start-Round-Robin algorithm in Figure 5.38 are the worst results compared to the results of both Deterministic-Coordinated and Randomised-Local versions of Reduce-Max when 2, 4 and 8 clusters are applied. However, when one and 16 clusters are in the system, the results are better than the best

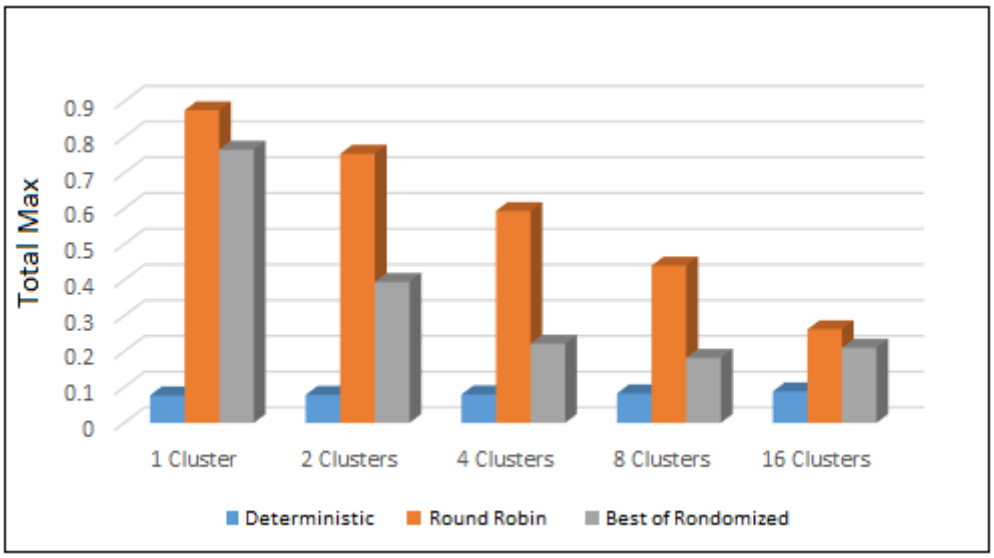


Figure 5.37: Total-max of Random-Start-Round-Robin algorithm for Poisson (2) weights

results of Randomised-Local version of Reduce-Max algorithm and worse than Deterministic-Coordinated results.

Random-Start-Round-Robin result for one Cluster is 0.47. This value of total-max is falling to the value of 0.15 when 16 clusters are in the system. The best result of Randomised-Local version of Reduce-Max starts from about 0.72 for one cluster and ends with around 0.16 for 16 clusters. However, the results of Deterministic-Coordinated version that keeps giving the best results start from around 0.074. The results rise until reaching 0.08 when 16 clusters are applied.

### 5.6.6 Conclusions

In this section dedicated to Random-Start-Round-Robin algorithm, there is an attempt to compare it with Reduce-Max algorithm and its two versions in hierarchical systems. Overall, the results of Random-Start-Round-Robin algorithm is usually better than the best results of Randomised-Local Reduction. However, Deterministic-Coordinated version of Reduce-Max, as usual, gives the best results that could be called the optimal results.

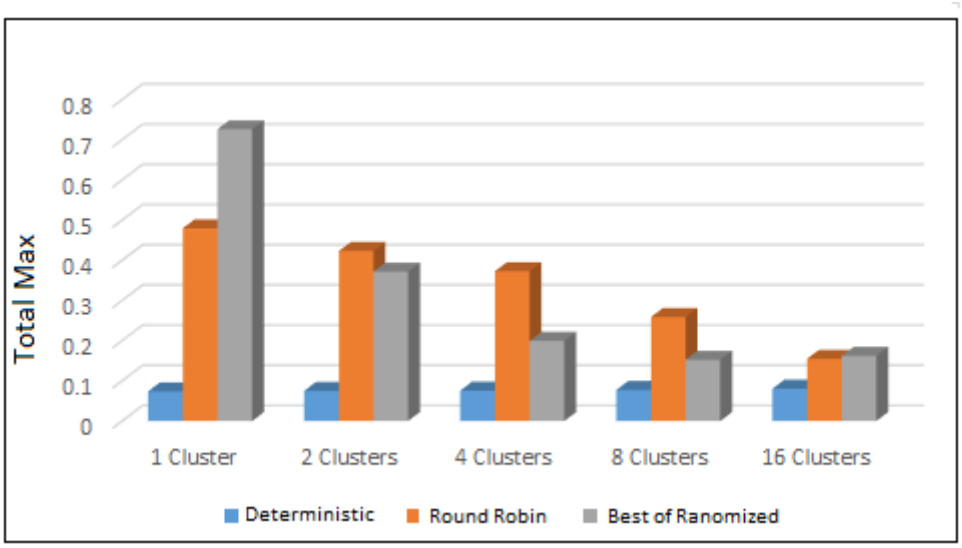


Figure 5.38: Total-max of Random-Start-Round-Robin algorithm for Poisson (8) weights

## 5.7 Balancing Mobility algorithm for monitoring Virtual Machines

In this section, we give and discuss the results of simulations for each set of input weights for Balancing Mobility algorithm.

In the resulting graphs below, the fifteen columns of different coloured boxes display maximum-weight of Reduce-Max in which FVMs choose VMs using Randomised Local reduction from the range of 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 and 16 of highest weight VMs, respectively. Sixteen rows correspond to different numbers of FVMs - from 1 to 16. The red boxes - one for each considered number of FVMs - denote the boxes with the smallest maximum weight among the results for random selection from 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 and 16 highest latency VMs, i.e. indicate the best highest range (i.e., column id) to be used by the Randomised Local reduction for each considered number of FVMs (row).

### 5.7.1 Random weights

Results in Figure 5.39 show that the more FVMs we apply for Randomised-Local Reduction, the wider range of highest values achieves the best total-max. This conclusion comes from the observation that the red box for one FVM indicates the range of two highest values as the best

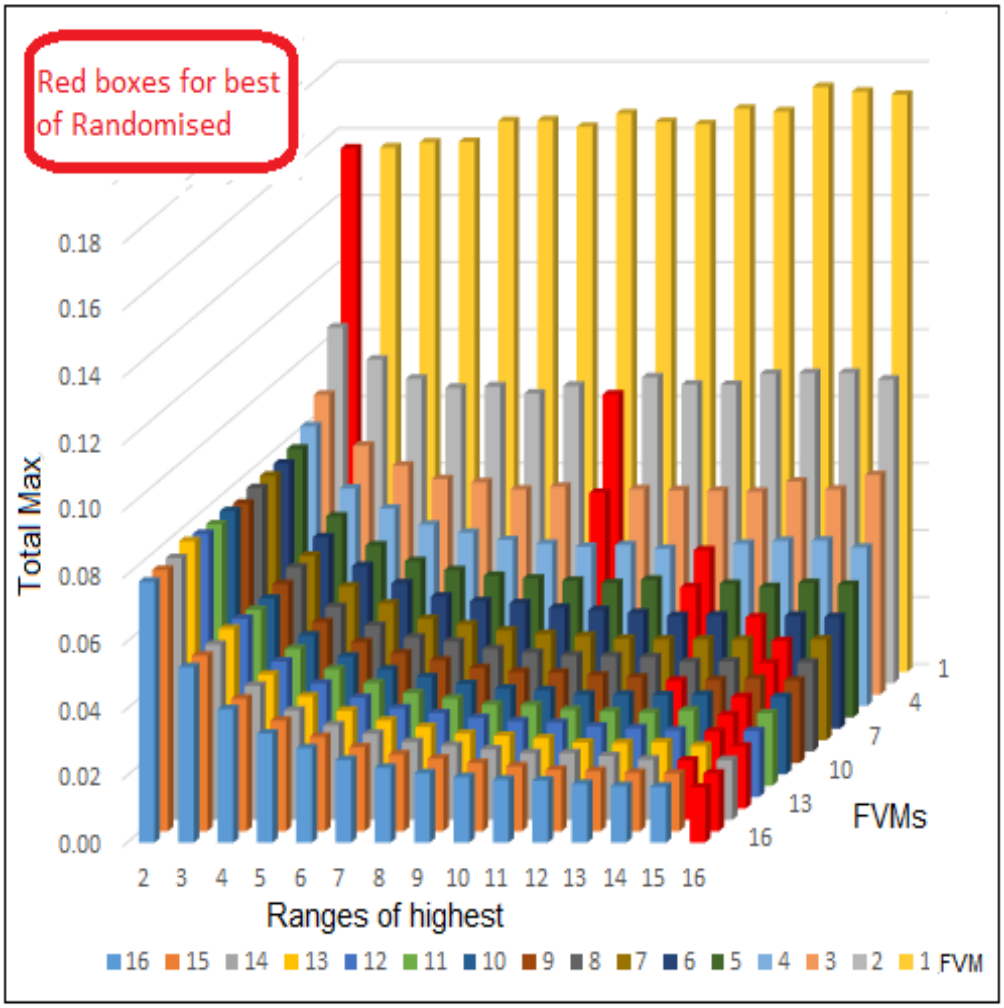


Figure 5.39: The total-max results for Randomised-Local version of Reduce-Max algorithm, taken for fifteen settings of the highest range, for balancing Mobility algorithm applied to Random weights

for Randomised local Reduction; however, the red box moves towards the range of 11 highest values for 2 FVMs and it is in the range of 16 highest values for 12, 14 and 15 FVMs.

Additionally, from Figure 5.39, for only one FVM we get worse total-max weight for a wider range of highest values. It is around 0.16 for the window of two highest and just under 0.18 for the window of 16 highest values. However, these results change with the increase of the number of FVMs. For example, the total-max is about 0.08 for the range of two highest values if there are 16 FVMs, then it drops down to just above 0.02 for the range of 16 highest values.

From the above, the highest total-max for random distribution is around 0.18. The red

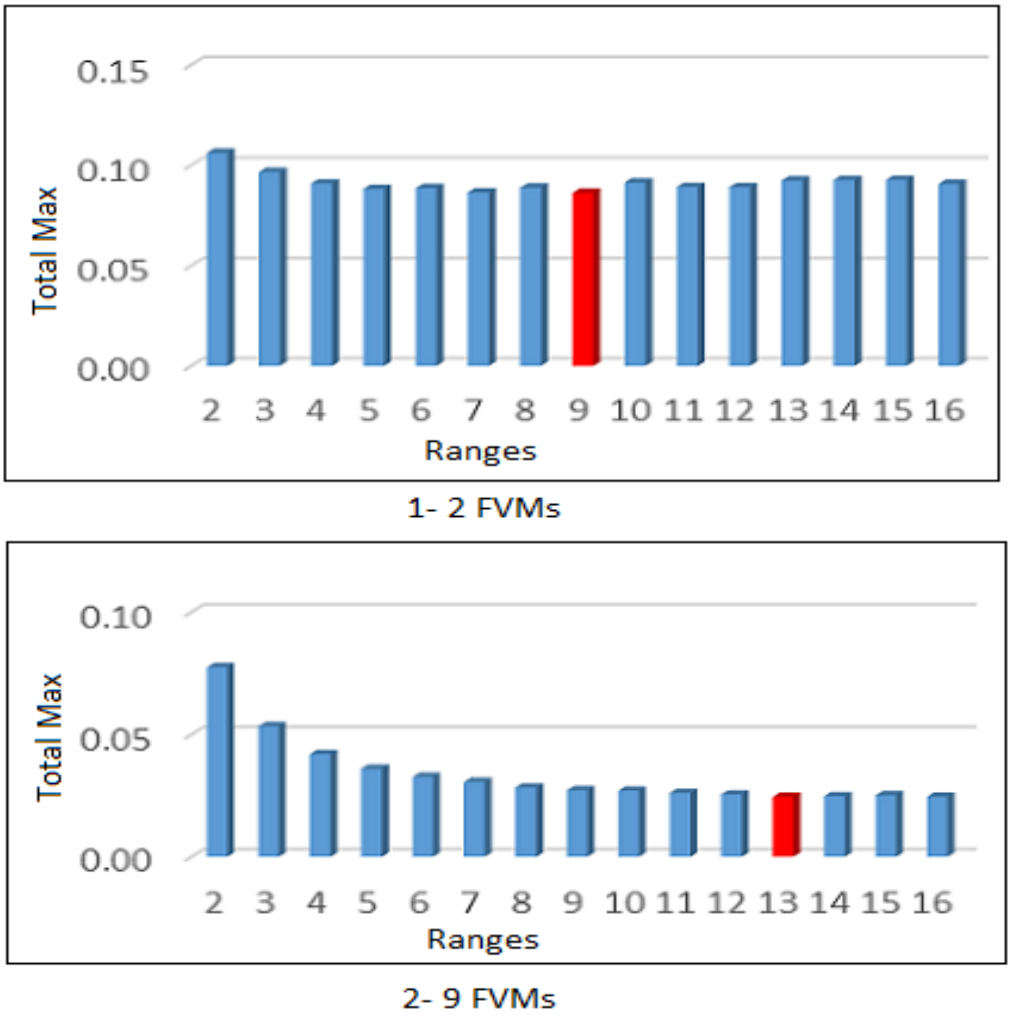


Figure 5.40: Samples of the total-max results for 2 and 9 FVMs taken for fifteen settings of the highest range for Balancing Mobility applied to Random weights

boxes reach the range of 16 highest if only 14 FVMs are applied in the system.

In Figure 5.40, two slides are taking from Figure 5.39 to see a deep view into the graph. In this figure, it can be seen clearly that the highest range can be reached when 2 FVMs are applied in the system, is 9 highest. However, the best range is 13 highest when 9 FVMs are applied.

### 5.7.2 Uniform weights

Unlike random weights, for uniform weights the red box, showing the best range for choosing a random VM from, changes from the range of 2 highest through the range of 9 and 13 highest

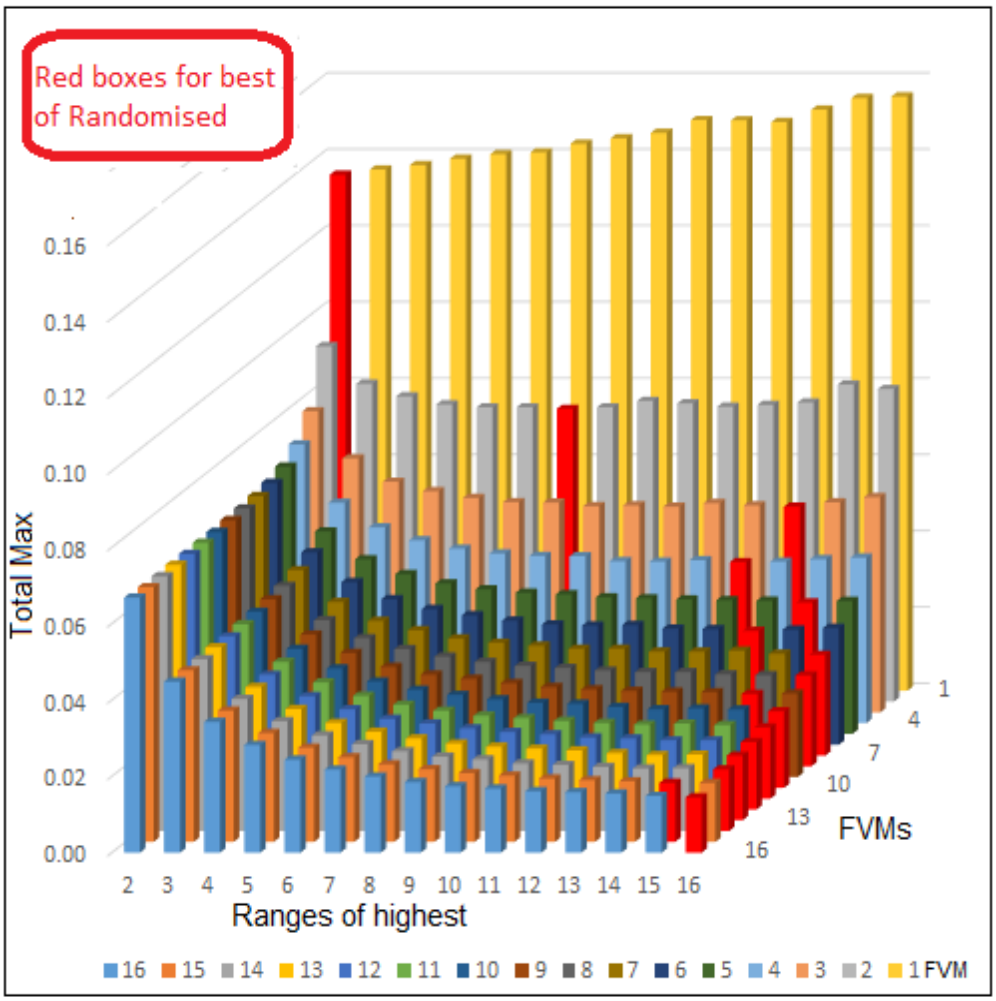


Figure 5.41: The total-max results for Randomised-Local version of Reduce-Max algorithm, taken for fifteen settings of the highest range, for balancing Mobility algorithm applied to Uniform weights

until it reaches 16 highest range for 11, 13, 14, 15 and 16 FVMs.

The Uniform distribution weights, as in Figure 5.41, show better results than the Random distribution in regards to two aspects. The first one is about the number of red boxes in the range of 16. The second one is about the total-max itself. The highest total-max for Uniform set of weights is just under 0.16, however, it is around 0.18 in the case of random weights.

The two slides in Figure 5.42 are taken from Figure 5.41 with the same numbers of FVMs to compare among all three sets of weights and to have a deeper look inside the graphs. Here for Uniform set, it can be clarified that the highest range can be reached when 2 FVMs are applied in the system, is 8. It was 9 for Random set of weights. In addition, 15 highest

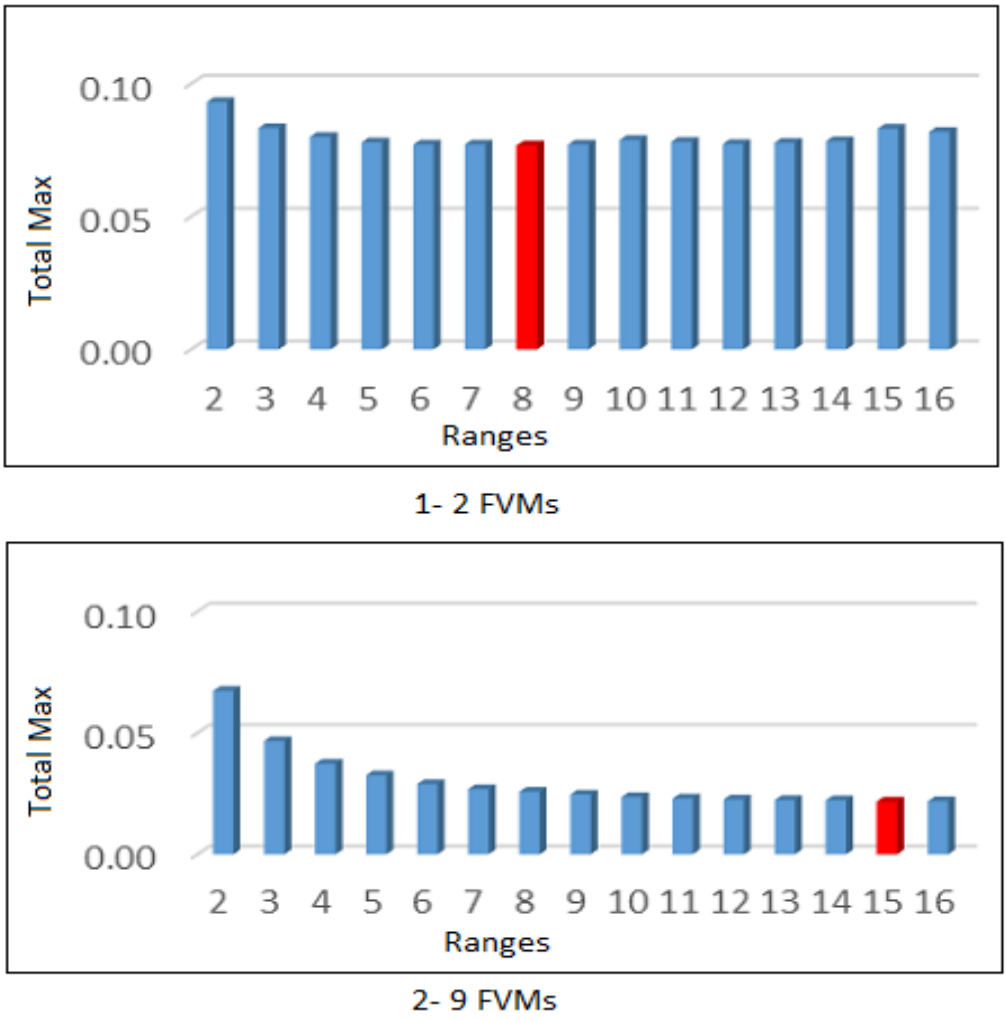


Figure 5.42: Samples of the total-max results for 2 and 9 FVMs taken for fifteen settings of the highest range for Balancing Mobility applied to Random weights

range is reached when 9 FVMs are applied.

### 5.7.3 Poisson distribution weights

The pattern of red boxes in choosing the best range of highest values for randomised-Local Reduction for Poisson distribution, cf., Figure 5.43 , is different from what was shown previously for Random weights. There are 12 red boxes out of 16 in the top two ranges with 15 and 16 highest ranges. Nine of them are in the highest range in the system. The red boxes are moving from the range of 2 highest through the range of 11 and towards the range of 15 and 16 highest for 1, 2, 3, and 4 towards 16 FVMs respectively. The highest total-max for Poisson distribution is the smallest. However, it is similar to the one for Uniform sets.

The two slides in Figure 5.44 are taken from Figure 5.43 for the same numbers of FVMs, 2 and 9 FVMs. For Poisson distribution, the highest range can be reached when 2 FVMs are applied in the system is 12. Additionally, when 9 FVMs applied for Poisson distribution, it is in the 16 highest range that is the biggest range the system.

### 5.7.4 Conclusions

In this section of Balancing Mobility algorithm, we attempted to quantify the belief that any malicious behaviour of cybercriminals in the cloud could be detected early. We focus on a simplified version of the mobility algorithm, called the Reduce-Max algorithm, and the version using Randomised-Local Reductions, which discover the symptoms of malicious behaviour. Three configurations of distribution of weights were considered, implemented and analysed for the Reduce-Max algorithm. Overall, the best results for Randomised-Local Reduction, which is the version of Reduce-Max requiring little resources and no coordination, occur in the case of Poisson and Uniform set of weights. The results for random weights are less efficient for most cases of applying FVMs than the other two considered distribution.

## 5.8 Summary of the chapter

This long chapter gave all results of experiments for all the proposed algorithms starting with Reduce-Max algorithm through all other algorithms. From this chapter, in homogeneous sys-

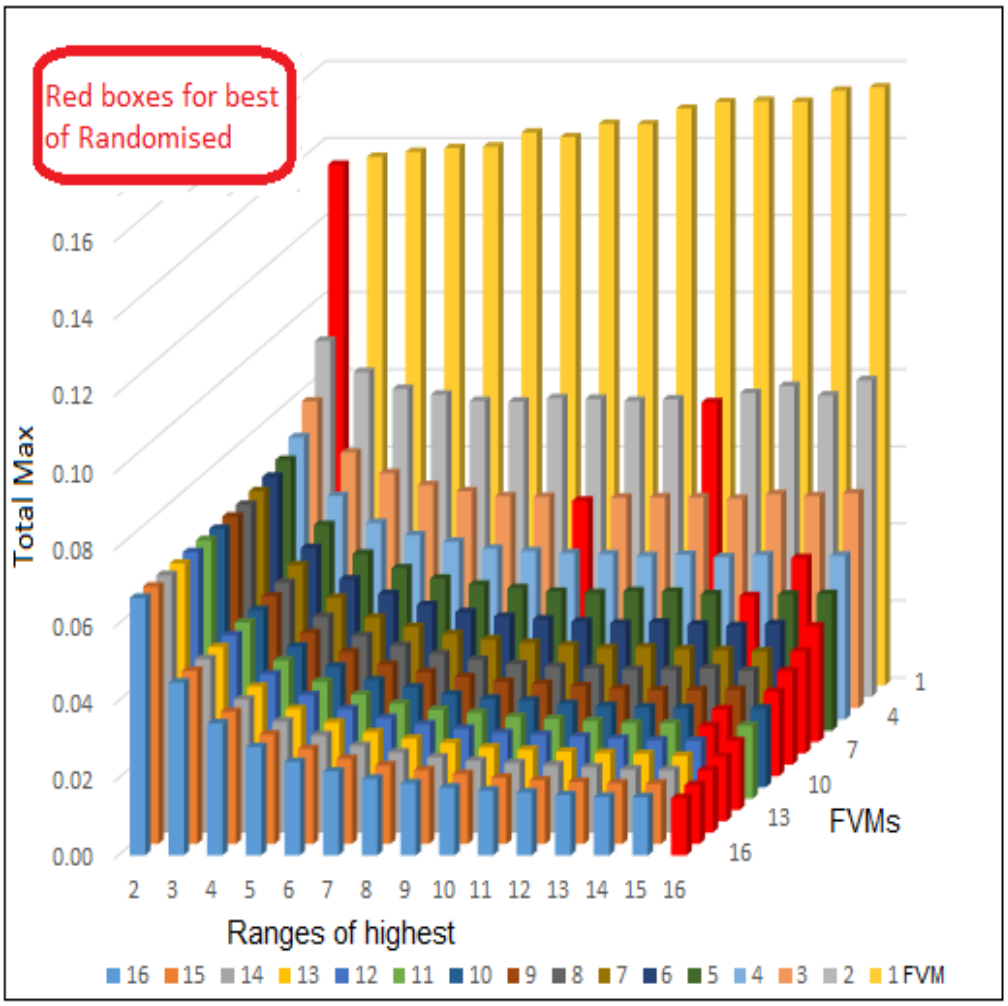


Figure 5.43: The total-max results for Randomised-Local version of Reduce-Max algorithm, taken for fifteen settings of the highest range, for balancing Mobility algorithm applied to Poisson weights

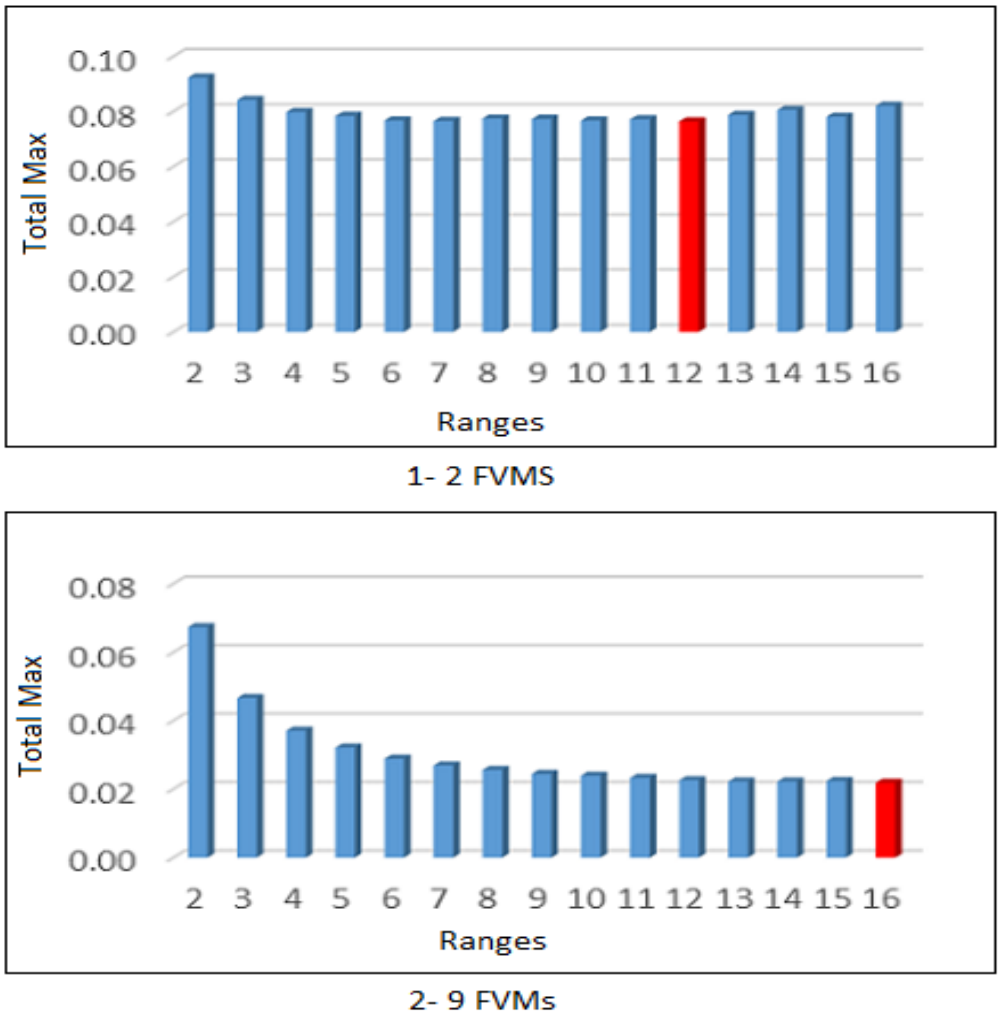


Figure 5.44: Samples of the total-max results for 2 and 9 FVMs taken for fifteen settings of the highest range for Balancing Mobility applied to Poisson weights

tems, the two versions of Reduce-Max algorithm are discussed as follows. Firstly, for Uniform, Random and Arithmetic weights the differences between the best results of Randomise-local version and the results of Deterministic-Coordinated version are not big. However, there is a big difference between the two versions in the case of Harmonic and Exponential weights. In hierarchical systems, there is little difference between Deterministic-Coordinated results and the best results of Randomised-Local with a bigger number of clusters.

For Random-Start-Round-Robin algorithm, it gave better results comparing to Randomised-Local version of Reduce-Max algorithm in most cases. The reason could be that FVMs in the case of Random-Start-Round-Robin algorithm reduce each VM weight one in $N$ rounds. This is probable reason why its results are better. However, the results of Deterministic-Coordinated Reduction are the best comparing to other algorithms. The reason is that deterministic version of Reduce-Max hits the highest latencies in each round.

Regarding data setting, in general, Uniform and Poisson (8) weights have the best results among all others data sets, but Random ([1,3] and [1,15]) and Poisson (2) weights were not far from them. Finally, Arithmetic come afterwards, whereas Harmonic and Exponential gave the worst results. The next chapter will contain discussions of these results of algorithms.

# Chapter 6

# Discussions and Evaluations

## 6.1 Overview

This chapter is about the discussions and evaluations of all algorithms. It depends on the previous chapter of experiment results. This chapter is organised as in the following two paragraphs. Section 6.2 presents the comparisons of the algorithms that were presented in this thesis. The reason for starting with comparisons is to have thorough discussions about all proposed algorithms. This section includes three subsections. First one is a comparison between Deterministic-Coordinated and Randomised-Local versions of Reduce-Max algorithm in both homogeneous and hierarchical systems. The second one shows a comparison between Reduce-Max algorithm and Random-Start-Round-Robin algorithm. The last one presents a comparison between hierarchical-structure and homogeneous structure. Section 6.3 provides important discussions about each data set. It includes six subsections. These subsections will have a separate discussion on each of them. Then, Section 6.4 contains discussions about the ranges of highest VMs and the numbers of FVMs. Finally, a summary of this chapter is presented in Section 6.5.

## 6.2 Comparison of algorithms for discussions

As explained earlier, this section contains comparisons of algorithms proposed in this thesis.

### 6.2.1 Comparison between Deterministic Coordinated version and Randomised Local version

The two versions of Reduce-Max algorithm that are proposed by the author of this thesis, have been used in most sections of the thesis as an essential part of it. The advantage of Deterministic-Coordinated version is that it hits the highest value. Therefore, it could have better results. However, the advantage of Randomised-Local version is that it has lightweight implementation, i.e. not relying on substantial resources or coordination and communication. Next, we present a comparison between these two variants in both studied structures.

#### 6.2.1.1 Homogeneous-structure

In homogeneous systems, the two versions of Reduce-Max algorithm are discussed as follows.

Firstly, for Uniform, Random and Arithmetic weights in Figures 5.12, 5.10 and 5.14 respectively, the differences between the best results of Randomise-local version and the results of Deterministic-Coordinated version are not big, the ratio between them does not exceed 1.8. Here are some examples of the results. For Uniform weights, the best result of Randomised-Local version among all highest ranges for 5 FVMs is just above 0.26, whereas the result of Deterministic-Coordinated version for the same number of FVMs is just above 0.2. Therefore, it can be seen that the difference is not big.

Similarly, for Random and Arithmetic weights, the best result of Randomised-Local for Random weights when 9 FVMs were applied is about 0.18 and the Deterministic-Coordinated result is around 0.13. For Arithmetic, best result of Randomised-Local is around 0.2, and the result of Deterministic-Centralised is just over 0.11. However, there is a big difference between the two versions in the case of Harmonic and Exponential weights, as shown in Figures 5.16 and 5.18. The ratio between the two versions reaches 6 in the case of Harmonic weights and 11 in the case of Exponential weights.

From previous paragraphs, the use of Randomised-Local type with its great benefits towards saving time in coordination and communication depends on the use of data sets. For Uniform, Random and Arithmetic weights the difference is acceptable. However, for other weights is not.

#### 6.2.1.2 Hierarchical-structure

In hierarchical system, both versions of Reduce-Max are studied. The comparison between them will be given according to data sets:

- Uniform weights.

  For Uniform weights, in the case of 16 FVMs applied in Figure 5.24, there is little difference between Deterministic-Coordinated results and the best results of Randomised-Local with a bigger number of clusters. When 16 clusters are in the system, deterministic result is 0.062 whereas best result of randomised version is about 0.13.

  However, the difference is bigger with smaller number of clusters as can be seen in Figures 5.24 and 5.25. The reason is that the range for bigger numbers of clusters is smaller with high possibility of hitting the highest value. For smaller numbers of clusters, on the other hand, the range is bigger.

- Poisson (2) weights.

  When Poisson weights with $\mu = 2$ are applied cf., Figure 5.30, the difference is smaller between the two version for bigger numbers of clusters. In the case of 16 clusters, the ratio between the best result of Randomised-Local reduction and Deterministic-Coordinated result is about 2, see Figure 5.31. This ratio is about 10 in the case of one cluster. This leads to conclusion that more clusters are better for applying Randomised-Local reduction.

- Poisson (8) weights.

  When Poisson weights with $\mu = 8$ are applied cf., Figure 5.32, the difference trend is similar to Uniform and Poisson (2) trends. The ratio between the two versions with smaller number of clusters is big, whereas it is around 2 for 16 clusters. In the case of one cluster, the best result of Randomised-Local reduction is around 0.75 while Deterministic-Coordinated result is about 0.073.

- Random [1,3] weights.

  For Random weights in range [1,3] in both Figures 5.26 and 5.27, the situation is not different. The differences between the two versions of Reduce-Max algorithm is acceptable for bigger numbers of clusters. With smaller numbers of clusters the difference is wider and could reach 10.

- Random [1,15] weights.

  The same trend of differences between the two versions appeared in Figure 5.28 when Random [1,15] weights were applied. The best result of Randomised-Local reduction is about 0.16 when 16 clusters were applied, while Deterministic-Coordinated result is only approximately 0.083. For one cluster, on the other hand, the best result of Randomised-Local reduction is about 0.76, whereas deterministic result is 0.075.

#### 6.2.1.3 Summary

In this brief conclusion, the comparison between the two versions of Reduce-Max is to meet the main objective of choosing algorithms to reach our goal of visiting VMs to discover threats symptoms. Overall, the differences between the best results of Randomised Local reduction and Deterministic Coordinated reductions in most cases are small in both variants of structures. However, for smaller numbers of clusters, in hierarchical systems for all considered weights, the results indicate that the difference could be big. In addition, there also is a big difference between the two versions in the case of Harmonic and Exponential weights in homogeneous systems. Finally, according to system designer needs, any of the two variants could be used as a way of checking VMs.

### 6.2.2 Comparison between Reduce-Max algorithm and Random-Start-Round-Robin algorithm

To fulfill the main objective of the best algorithms of checking VMs to be chosen, here is a comparison between Reduce-Max and Random-Start-Round-Robin algorithms. To do so, two subsections below compare Random-Start-Round-Robin algorithm with each variant of Reduce-Max algorithm as follows:

#### 6.2.2.1 Deterministic-Coordinated Reduction

From previous comparisons, Deterministic-Coordinated reduction has better results compared to Randomised-Local reduction. Here, from Figures 5.34, 5.37, 5.38, 5.35 and 5.36, it can be clearly seen that the results of Random-Start-Round-Robin algorithm is worse than Deterministic-Coordinated reduction results. However, the difference between the two algorithms in the case

of Uniform weights in Figure 5.34 is small. For example, the results of Random-Start-Round-Robin algorithm is about 0.063 when 16 clusters were applied. Deterministic-Coordinated result is around 0.062 in the same setting. It means in this case, they are really close to each other. Second case is for Random [1,3] weights and when 16 clusters were applied cf., Figure 5.35, the difference is small too. For other types of weights, the difference is wider but not critical.

#### 6.2.2.2 Randomised-Local Reduction

Randomised-Local Reduction has an advantage of lightweight implementation, i.e., not relying on substantial resources or coordination and communication. However, from Figures 5.34, 5.35 and 5.36 for Uniform and Random [1,3] and [1,15] weights, it is clear that the results of Random-Start-Round-Robin algorithm are better than the results of Randomised-Local reduction. However, for Poisson (2) weights in Figure 5.37 and for clusters 2, 4, 8 of Poisson (8) weights in Figure 5.38, the results of Randomised-Local reduction are better.

#### 6.2.2.3 Summary

The previous comparisons of algorithms lead us to say that Random-Start-Round-Robin algorithm gives better results comparing to Randomised-Local version of Reduce-Max algorithm in most cases. The reason could be that FVMs in the case of Random-Start-Round-Robin algorithm reduce all VMs weights in $N$ rounds. This is probable reason why its results are better. However, the results of Deterministic-Coordinated reduction are the best comparing to other algorithms. The reason is that deterministic version of Reduce-Max hits the highest latencies in each round.

Overall, the results of Random-Start-Round-Robin algorithm are in the middle. They are better than the results of Randomised-Local reduction and worse than those for Deterministic-Centralised reduction.

### 6.2.3 Comparison between hierarchical-structure and homogeneous structure of Reduce-Max Algorithm

After algorithms comparisons and discussions, now it is needed to fulfill the related objective about cloud structures. Here is a comparison between both hierarchical and homogeneous

structures. Next, we will compare them in regards to the total-max when a number of FVMs is 16.

Firstly, homogeneous structure was studied for three problems as follows.

- Reduce-Max algorithm

  For Reduce-Max algorithm, the result of total-max for Deterministic-Coordinated when 16 FVMs were applied in the case of Uniform weights is 0.0625 cf., Figure 5.11. The best of Randomised-Local reduction with 16 FVMs is about 0.1112. Uniform weights are considered here because they demonstrated the best results of Reduce-Max algorithm.

- Symptoms appearance problem

  The best of Randomised-Local Reduction after assumption of symptoms appearance with 16 FVMs for Uniform weights is around 0.1471 cf., Figure 5.20.

- Balancing Mobility algorithm

  The best of Randomised-Local reduction with 16 FVMs for Uniform weights is approximately 0.1445 cf., Figure 5.41.

Secondly, hierarchical-structure was also studied in two problems in the thesis and the findings are as follows.

- Hierarchical model

  For hierarchical model, the results of Deterministic-Coordinated when 16 FVMs were applied in the case Uniform weights are around 0.062 for all numbers of clusters cf., Figure 5.24. Randomised-Local reduction, on the other hand, has the result varying from 0.64 for one cluster to 0.14 when 16 clusters were applied.

- Random-Start-Round-Robin algorithm

  When Random-Start-Round-Robin algorithm is used, the results of 16 FVMs for Uniform weights start from around 0.23 for one cluster and reach best value of 0.063 when 16 clusters applied c.f, Figure 5.34.

#### 6.2.3.1 Summary

From the previous discussion, it is clear that the results for 16 FVMs and for Uniform weights in both structures are close to each other. However, there are differences between Deterministic-Coordinated reduction and Randomised-Local reduction, and advantages of each. For instance, in the hierarchical model the trends in performance depend primarily on the number of clusters in all the considered settings.

## 6.3 Discussions about each data sets

The following subsections provide discussions about each of data sets.

### 6.3.1 Random weights

Random weights are studied for all considered issues in this thesis. They are studied for two different ranges [1,3] and [1,15]. Firstly, with original Reduce-Max in both Figures 5.2 and 5.3, the results show smooth trends and it was around 1.2. Then, with the two versions in Figure 5.9, the results were the second best after Uniform weights. The same happened with symptom appearance in Figure 5.19. After that, in hierarchical model, cf., Figures 5.26 and 5.28, the results of Random weights were worse than Uniform and Poisson with $\mu = 2$, but the difference was not critical. The same situation happened with Random-Start-Round-Robin algorithm and Balancing Mobility algorithm. Therefore, Random weights are good to be used in some cases and models.

### 6.3.2 Uniform weights

Uniform weights are studied for all considered algorithms in this thesis. Beginning with original Reduce-Max in Figure 5.1, they gave the best results due to the reduction of all weights in $N$ round. Moreover, it shows the best results for the two versions of Reduce-Max in Figure 5.11 and with symptom appearance in Figure 5.20. Then, for the hierarchical model and Random-Start-Round-Robin algorithm in both Figures 5.24 and 5.34, the results continue to indicate the best results. Again for Balancing Mobility algorithm, the results also show the best performance. Overall, Uniform weights indicates best results in most cases in regards to the total-max.

### 6.3.3 Arithmetic series weights

Arithmetic weights are studied for three algorithms. First is the original Reduce-Max in Figure 5.6. It shows a trend with result of almost 1.3 that is worse than Uniform and Random. The second case is the two version of Reduce-Max in Figure 5.13 the results here were worse than those for Uniform and Random too. Finally, with symptom appearance in Figure 5.21, the results were worse than for Uniform. However, it was close to the results for Uniform and Random weights.

### 6.3.4 Harmonic weights

Harmonic weights were studied for the same sittings as Arithmetic weights. They show better results for the original Reduce-Max in Figure 5.7. However, with the work of the two version of Reduce-Max and symptoms appearance, in Figures 5.15 and 5.22, they yield worse results.

### 6.3.5 Exponential weights

The results on Exponential weights were better in the case of the original Reduce-Max in Figure 5.8, but then Exponential weights show worse results for the two versions of Reduce-Max and symptoms appearance in both Figures 5.17 and 5.23. The results are even worse than for Harmonic weights.

### 6.3.6 Poisson (2) weights

In this thesis, Poisson (2) weights yield better results than at least Random [1,3] cf., Figure 5.4. These weights also continue to indicate good results for hierarchical model in Figures 5.30, and 5.37. However, Poisson (2) weights are not within the best results of Random-Start-Round-Robin algorithm.

### 6.3.7 Poisson (8) weights

Poisson (8) weights yield the second best results after Uniform weights cf., Figure 5.5. For hierarchical model, in Figures 5.32 and 5.38, Poisson (8) weights usually have the second best results after Uniform weights.

Finally, Poisson weights for Balancing Mobility algorithm show the best results along with Uniforms weights cf., Figure 5.43.

### 6.3.8 Summary

Overall, cloud designers could evaluate their algorithms for the data setting that suits their systems according the number of FVMs and the structure of cloud systems. However, in general, Uniform and Poisson (8) weights have the best results among all others data sets, but Random ([1,3] and [1,15]) and Poisson (2) weights were not far from them. Finally, Arithmetic come afterwards, whereas Harmonic and Exponential gave the worst results.

## 6.4 Discussions about the ranges of highest VMs and the numbers of FVMs

The highest ranges mean that the range of highest weighted VMs that FVMs choose randomly to visit then reduce its weights. This way of reduction is Randomise-Local version of Reduce Max. It has been studied for both homogeneous and hierarchical structures.

For homogeneous systems, the best results occur with increasing number of highest ranges. The reason is that the probability of choosing the same VM is low.

However, for Harmonic and Exponential weights, the best results are in ranges of 2 and 3. Therefore, the best results of total-max are in higher ranges with Uniform, Random[1,15] and Arithmetic weights, whereas they are in smaller ranges in the case of the weights that have worse results.

In hierarchical systems, the situation is different. The range of FVMs is multiplied by $\alpha$. We have in general better results for ranges that have smaller values of $\alpha$, specifically when $\alpha = 1$. The reason is that the probability of choosing the highest weighted VM is bigger with smaller value of $\alpha$.

## 6.5 Summary of the chapter

This discussion chapter gave evaluations of all proposed algorithms and discussions about their results. It started with a comparison between the two versions of Reduce-Max in both structures. The differences between them in most cases are small in both variants of structures. However, for smaller numbers of clusters, in hierarchical systems the differences are bigger. In addition, there also is a big difference between the two versions in the case of Harmonic and Exponential weights in homogeneous systems. Then the chapter moves to discuss the comparison between Reduce-Max and Random-Start-Round-Robin algorithms. The results of Random-Start-Round-Robin algorithm are better than the results of Randomised-Local reduction and worse than those for Deterministic-Centralised reduction. After that, this chapter had a section of comparing homogeneous systems with hierarchical systems. The conclusion about the hierarchical model is that the trends in performance depend primarily on the number of clusters. Finally, a section was about data sets. That section contained an evaluations of each consider weights set.

# Chapter 7

# Conclusions and Future Work

## 7.1 Overview

This ending chapter is the last chapter of this thesis. It will end the thesis by providing a summary of all previous chapters including main findings and results.

The sections in this chapter are organised as follows. Section 7.2 considers a research contribution that described what was done in this thesis. It has six subsections. This section is followed by the main results and findings. Section 7.4 discusses the future work of this research. This section includes the future directions as follows: cluster shifting, classifications of symptoms and some more complex configuration of symptoms.

## 7.2 Research Contribution

The main aim of this research was to design and evaluate new algorithms for distributed monitoring of cloud systems. These algorithms target efficient discovery of symptoms of malicious behaviours in cloud systems.

### 7.2.1 Reduce-Max algorithm

Firstly, a simplified version of Mobility algorithm called Reduce-Max algorithm was introduced. Eight configurations of weights were considered, implemented and analysed for

Reduce-Max algorithm. For the work of a single type of FVMs, Uniform weights, through a different number of VMs, achieves the best results (the lowest total-max).

### 7.2.2 Two variants of reductions for Reduce-Max for many FVMs

Then two versions of reductions for the original Reduce-Max algorithm were introduced. They are Deterministic-Coordinated reduction and Randomised-Local reduction.

The best results for randomised local reduction, which is the version of Reduce-Max requiring little resources and no coordination, occur in the case of Uniform weights. However, the results of Random weights and Arithmetic series weights are close to those for Uniform weights. The results for Harmonic and Exponential weights was indicated as the worst even in the case of Deterministic-Coordinated reduction.

### 7.2.3 Reduce-Max and symptom appearance

Moreover, an attempt to show that Reduce-max algorithm behaves well with symptoms' appearance. The reason is that discovering symptoms can help to detect malicious behaviour of cybercriminals in clouds earlier. The results indicate that the execution of Reduce-Max algorithm is behaving well compared to the results of the two versions of Reduce-Max algorithm for Random, Uniform and Arithmetic weights. However, there is a similarity to the results presented previously for Harmonic and Exponential weights.

### 7.2.4 Hierarchical topology model

The work was expanded to examine the impact of hierarchical structure from the perspective of cloud monitoring. The experiments were done to compare the results for different numbers of clusters, including one cluster that corresponds to previously considered homogeneous systems, under the same number of FVMs. three distributions were used to model weights of VMs: Uniform, Random and Poisson, and considered different ranges of highest range for random selection of next VM according to the value of $\alpha$.

From the simulations results, the use of hierarchical structure is recommended because it is effective with all the considered distributions. In all cases, the results are better when the

number of clusters is increasing, although, generally, the results for 8 clusters are similar to 16 clusters. In regards to the sets of weights, Uniform weights, followed by Poisson with $\mu = 8$, are more tractable than other weights, although the differences are not critical.

For the question of best choice of highest ranges to implement random reduction, we get in general better results for smaller values of $\alpha$, especially $\alpha = 1$. Importantly, the trends in performance depend primarily on the number of clusters in all the considered settings. This confirms the importance of proper structuring of the cloud for the efficiency of the monitoring process, in particular, for implementing light security and resiliency systems on top of the cloud.

### 7.2.5 Random-Start-Round-Robin algorithm

For Random-Start-Round-Robin algorithm, there is an attempt to compare it with Reduce-Max algorithm by its two versions in hierarchical systems. Overall, the results of Random-Start-Round-Robin algorithm is usually better than the best results of Randomised-Local reduction. However, deterministic version of Reduce-Max, as usual, give the best results that could be called the optimal results.

### 7.2.6 Balancing Mobility algorithm for monitoring Virtual Machines

Three configurations of distribution of weights were considered, implemented and analysed for Balancing Mobility algorithm.

Overall, the best results for Randomised-Local reduction occur in the case of Poisson weights and Uniform set of weights. The results for Random weights are less efficient for most cases of applying FVMs than the other two considered sets of weights.

## 7.3 Main Results and Findings

The main objective of this thesis was to propose new algorithms to achieve the target of efficient discovering malicious behaviours' symptoms. The findings about these algorithms are listed as follows:

- The comparison between the two versions of Reduce-Max is to meet the main objective of which algorithms to choose to reach our goal of efficiently visiting VMs to discover threat symptoms. Both variants have advantages: the deterministic version has the best total-max results in all experiments and the randomised version requires little resources and no coordination and its performance is not very far from Deterministic-Coordinated results. The differences between the best results of Randomised-Local reduction and Deterministic Coordinated reduction in most cases are small in both kinds of structures homogeneous and hierarchical. The use of Randomised-Local variant with its great benefits toward saving time in coordination and communication in homogeneous systems depends on the use of data sets. For Uniform, Random and Arithmetic weights, the differences are small. However, for other set of weights, it is worse.

- Random-Start-Round-Robin algorithm gives usually better results than the best achieved by the Randomised-Local reduction. However, the deterministic version of Reduce-Max, as usual, gives the best results. The reason could be that FVMs in the case of Random-Start-Round-Robin algorithm reduce all VMs weights in $N$ rounds.

- If there is only one cluster in the system, both structures have same results. However, there are differences between them and advantages of each. For instance, in the hierarchical model, the trends in performance depend primarily on the number of clusters in all the considered settings.

- Cloud designers could choose the data setting that suits their systems according to the number of FVMs and the way cloud system is modelled. However, generally as this work shows, Uniform firstly then Poisson (8) weights guarantee the best performance among all others data sets, but both Random weights were not that far from them. Finally, Arithmetic weights were in the middle, whereas Harmonic and Exponential weights gave the worst results.

- For the homogeneous system, the best results are for bigger ranges with Uniform, Random and Arithmetic weights. However, the best total-max is for smaller ranges in the case of Harmonic and Exponential weights.

- In general, for hierarchical systems, the best results are for ranges that have smaller values of $\alpha$.

## 7.4 Future Work

In this section, there is a presentation of promising directions for future work. Potential future directions can be itemised as follows.

- Algorithms

  From this thesis, an interesting direction of future work could be about more investigations for the algorithms that are studied in this work or have more algorithms for the optimal solution of the main issue.

- Comparison between the proposed algorithms and the Bamboo Garden paper

  In the paper titled "Bamboo Garden Trimming Problem (Perpetual maintenance of machines with different attendance urgency factors)"[22], the authors proposed a solution to a scheduling problem. They have an algorithm that is similar to Reduce-Max. However, their algorithm did not reduce the max value. Instead, it waits until values reach some given thresholds and they reduce the fastest growing one.

  This idea of waiting until reaching the thresholds could increase the total-max. The second dough about this work is that their method of hitting the fastest growing does not guarantee that FVMs visits all VMs in short time "rounds" which gives the best total-max. This is the power of the proposed algorithms especially the deterministic version of Reduce-Max algorithm that hits the max value in each round.

  Finally, it can be said that a interesting direction is to have experiments for Bamboo Garden paper and compare results of proposed algorithms with it and discuss them for the best total-max.

- Data sets

  Another direction could be in the area of data settings. More data configurations could be suggested or more studies regarding the current ones.

- Cluster shifting

  From the work of hierarchical structure in this thesis, it can be stated that FVMs are distributed equally. For example, if the system has 16 FVMs and with 8 clusters, each cluster will have 2 FVMs. The reason was to examine the efficiency of the hierarchical model.

A promising direction here is a dynamic environment for FVMs among clusters. The meaning is that FVMs could move from a cluster to another cluster inside the gateway. For instance, if one cluster is fine with a good total-max, someFVMs can move from this cluster to another cluster that has issues with the total-max.

- Classification of symptoms

  It has been known from this thesis that the studied algorithms are to detect malicious behaviours' symptoms. The work with symptoms could be another interesting future extension of this research. An example of that is to do classification of symptoms according to attacks. For instance, if an attack results in four symptoms, these four symptoms could be grouped into one class. Note that, different attack could result in the same symptoms.

# Bibliography


[1] Sultan Alshamrani, Dariusz Kowalski, and Leszek Gasieniec. The impact of hierarchical structure on efficiency of cloud monitoring. *Recent Patents on Computer Science Journal*, 2016.

[2] Sultan Alshamrani. Discovering malicious behaviour symptoms in cloud systems. In *Proceedings of the Eighth Saudi Students Conference in the UK*, pages 375–384. World Scientific, 2016.

[3] Sultan S. Alshamrani, Dariusz R. Kowalski, and Leszek A. Gasieniec. Efficient discovery of malicious symptoms in clouds via monitoring virtual machines. In *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on*, pages 1703–1710, Oct 2015.

[4] Sultan Alshamrani, Dariusz Kowalski, Leszek Gąsieniec, and Muhammed Abdulazeez. Balancing mobility algorithm for monitoring virtual machines in clouds. In *ECCWS2016-Proceedings fo the 15th European Conference on Cyber Warfare and Security*, page 9, 2016.

[5] Sultan Alshamrani, Dariusz Kowalski, and Leszek Gasieniec. The impact of hierarchical structure on efficiency of cloud monitoring. In *Cloud Computing Technologies and Applications (CloudTech), 2016 2nd International Conference on*, pages 40–46. IEEE, 2016.

[6] Sultan Alshamrani. How reduce max algorithm behaves with symptoms appearance on virtual machines in clouds. In *Cloud Computing (ICCC), 2015 International Conference on*, pages 1–4. IEEE, 2015.

[7] Dariusz R. Kowalski Sultan S. Alshamrani and Leszek A. Gasieniec. Random-start-round-robin algorithm for monitoring vms in clouds. 2017.

[8] Peter Mell and Tim Grance. The NIST definition of cloud computing. *National Institute of Standards and Technology*, 53(6):50, 2009.

[9] Lizhe Wang, Jie Tao, Marcel Kunze, Alvaro Canales Castellanos, David Kramer, and Wolfgang Karl. Scientific cloud computing: Early definition and experience. In *HPCC*, volume 8, pages 825–830, 2008.

[10] Yashpalsinh Jadeja and Kirit Modi. Cloud computing-concepts, architecture and challenges. In *Computing, Electronics and Electrical Technologies (ICCEET), 2012 International Conference on*, pages 877–880. IEEE, 2012.

[11] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, and Ion Stoica. A view of cloud computing. *Communications of the ACM*, 53(4):50–58, 2010.

[12] Subashini Subashini and Veeraruna Kavitha. A survey on security issues in service delivery models of cloud computing. *Journal of network and computer applications*, 34(1):1–11, 2011.

[13] Dimitrios Zissis and Dimitrios Lekkas. Addressing cloud computing security issues. *Future Generation computer systems*, 28(3):583–592, 2012.

[14] Meiko Jensen, Jörg Schwenk, Nils Gruschka, and Luigi Lo Iacono. On technical security issues in cloud computing. In *2009 IEEE International Conference on Cloud Computing*, pages 109–116. IEEE, 2009.

[15] Tharam Dillon, Chen Wu, and Elizabeth Chang. Cloud computing: issues and challenges. In *2010 24th IEEE international conference on advanced information networking and applications*, pages 27–33. IEEE, 2010.

[16] Asaf Shabtai, Robert Moskovitch, Yuval Elovici, and Chanan Glezer. Detection of malicious code by applying machine learning classifiers on static features: A state-of-the-art survey. *Information Security Technical Report*, 14(1):16–29, 2009.

[17] Przemyslaw Kazienko and Piotr Dorosz. Intrusion Detection Systems (IDS) Part I-(network intrusions; attack symptoms; IDS tasks; and IDS architecture). *Web paper available from http://techgenix.com/:{Nov. 2008}*, 2004.

[18] Keith Harrison, Behzad Bordbar, Syed TT Ali, Chris I Dalton, and Andrew Norman. A framework for detecting malware in cloud by identifying symptoms. In *Enterprise Distributed Object Computing Conference (EDOC), 2012 IEEE 16th International*, pages 164–172. IEEE, 2012.

[19] Hongyan Jing, Regina Barzilay, Kathleen McKeown, and Michael Elhadad. Summarization evaluation methods: Experiments and analysis. In *AAAI symposium on intelligent summarization*, pages 51–59, 1998.

[20] Peter Bauer and K Kohne. Evaluation of experiments with adaptive interim analyses. *Biometrics*, pages 1029–1041, 1994.

[21] Bonnie Kaplan and Joseph A Maxwell. Qualitative research methods for evaluating computer information systems. In *Evaluating the organizational impact of healthcare information systems*, pages 30–55. Springer, 2005.

[22] Leszek Gąsieniec, Ralf Klasing, Christos Levcopoulos, Andrzej Lingas, Jie Min, and Tomasz Radzik. Bamboo garden trimming problem (perpetual maintenance of machines with different attendance urgency factors). In *Proceedings of the 43rd International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2017)*, volume 10139. Springer Verlag, 2017.

[23] Muhammed Bello Abdulazeez, Dariusz Kowalski, Alexei Lisista, and Sultan Alshamrani. Failure or denial of service? a rethink of the cloud recovery model. In *ECCWS2016-Proceedings fo the 15th European Conference on Cyber Warfare and Security*, page 1.

[24] Corona Brezina. *Al-Khwarizmi: The inventor of algebra*. The Rosen Publishing Group, 2006.

[25] Donald E Knuth. The art of computer programming, volume 1: fundamental algorithms. redwood city. Redwood City, 1997.

[26] John A Simpson and Edmund SC Weiner. *Oxford English dictionary additions series*, volume 3. Oxford University Press, 1997.

[27] Michael R Gary and David S Johnson. Computers and intractability: A guide to the theory of NP-completeness, 1979.

[28] Lawrence Davis. Handbook of genetic algorithms. 1991.

[29] Carlos A Coello Coello, David A Van Veldhuizen, and Gary B Lamont. *Evolutionary algorithms for solving multi-objective problems*, volume 242. Springer, 2002.

[30] Claus-Peter Schnorr and Martin Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Mathematical programming*, 66(1-3):181–199, 1994.

[31] Heinz H Bauschke and Jonathan M Borwein. On projection algorithms for solving convex feasibility problems. *SIAM review*, 38(3):367–426, 1996.

[32] John R Koza. Non-linear genetic algorithms for solving problems by finding a fit composition of functions, August 4 1992. US Patent 5,136,686.

[33] Robert Endre Tarjan. Fast algorithms for solving path problems. *Journal of the ACM (JACM)*, 28(3):594–614, 1981.

[34] George F Coulouris, Jean Dollimore, and Tim Kindberg. *Distributed systems: concepts and design*. Pearson Education, 2005.

[35] Andrew S Tanenbaum and Maarten Van Steen. *Distributed systems*. Prentice-Hall, 2007.

[36] Ajay D Kshemkalyani and Mukesh Singhal. *Distributed computing: principles, algorithms, and systems*. Cambridge University Press, 2011.

[37] Andrew S Tanenbaum. *Distributed operating systems*. Pearson Education India, 1995.

[38] Christos H Papadimitriou. *Computational complexity*. John Wiley and Sons Ltd., 2003.

[39] Kai Li. IVY: A shared virtual memory system for parallel computing. *International Conference on Parallel Processing (ICPP'02)*, 88:94, 1988.

[40] Nancy A Lynch. *Distributed algorithms*. Morgan Kaufmann, 1996.

[41] David Peleg. Distributed computing: A locality-sensitive approach, siam. Technical report, ISBN 0-89871-464-8, 2000.

[42] Sukumar Ghosh. *Distributed systems: an algorithmic approach*. CRC press, 2014.

[43] Richard M Fujimoto. *Parallel and distributed simulation systems*, volume 300. Wiley New York, 2000.

[44] Joseph Carl Robnett Licklider and Welden E Clark. On-line man-computer communication. In *Proceedings of the May 1-3, 1962, Spring Joint Computer Conference*, pages 113–128. ACM, 1962.

[45] Barry M Leiner, Vinton G Cerf, David D Clark, Robert E Kahn, Leonard Kleinrock, Daniel C Lynch, Jon Postel, Larry G Roberts, and Stephen Wolff. A brief history of the internet. *ACM SIGCOMM Computer Communication Review*, 39(5):22–31, 2009.

[46] Charles Spurgeon. *Ethernet: the definitive guide*. " O'Reilly Media, Inc.", 2000.

[47] David I Gold. Internet gambling debt liability: Trouble ahead–a consideration of providian v. haines. *T. Jefferson L. Rev.*, 22:219, 1999.

[48] Lawrence G Roberts. Multiple computer networks and intercomputer communication. In *Proceedings of the First ACM symposium on Operating System Principles*, pages 3–1. ACM, 1967.

[49] Ian Peter. Ian peter's history of the internet, 2011.

[50] Greg R Andrews. *Foundations of parallel and distributed programming*. Addison-Wesley Longman Publishing Co., Inc., 1999.

[51] Rajkumar Buyya, Anton Beloglazov, and Jemal Abawajy. Energy-efficient management of data center resources for cloud computing: A vision, architectural elements, and open challenges. *arXiv preprint arXiv:1006.0308*, 2010.

[52] Renato J Figueiredo, Peter A Dinda, and José AB Fortes. A case for grid computing on virtual machines. In *Distributed Computing Systems, 2003. Proceedings. 23rd International Conference on*, pages 550–559. IEEE, 2003.

[53] Sander Pronk, Szilárd Páll, Roland Schulz, Per Larsson, Pär Bjelkmar, Rossen Apostolov, Michael R Shirts, Jeremy C Smith, Peter M Kasson, David van der Spoel, Berk Hess, and Erik Lindahl. Gromacs 4.5: a high-throughput and highly parallel open source molecular simulation toolkit. *Bioinformatics*, pages 1367–4803, 2013.

[54] Bala Iyer and John C Henderson. Preparing for the future: Understanding the seven capabilities cloud computing. *MIS Quarterly Executive*, 9(2), 2010.

[55] Qi Zhang, Lu Cheng, and Raouf Boutaba. Cloud computing: state-of-the-art and research challenges. *Journal of internet services and applications*, 1(1):7–18, 2010.

[56] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D Joseph, Randy H Katz, Andrew Konwinski, Gunho Lee, David A Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. Above the clouds: A berkeley view of cloud computing. 2009.

[57] Peter Mell and Tim Grance. The NIST definition of cloud computing. 2011.

[58] Qusay Hassan. Demystifying cloud computing. *The Journal of Defense Software Engineering*, pages 16–21, 2011.

[59] Rajkumar Buyya, Rajiv Ranjan, and Rodrigo N Calheiros. Modeling and simulation of scalable cloud computing environments and the cloudsim toolkit: Challenges and opportunities. In *High Performance Computing & Simulation, 2009. HPCS'09. International Conference on*, pages 1–11. IEEE, 2009.

[60] George Lawton. Developing software online with platform-as-a-service technology. *Computer*, 41(6):13–15, 2008.

[61] Sushil Bhardwaj, Leena Jain, and Sandeep Jain. Cloud computing: A study of infrastructure as a service (IaaS). *International Journal of engineering and information Technology*, 2(1):60–63, 2010.

[62] Bhaskar Prasad Rimal, Eunmi Choi, and Ian Lumb. A taxonomy and survey of cloud computing systems. *International Conference on Networked Computing, Advanced Information Management and Service and on Digital Content, Multimedia Technology and its Applications (INC, IMS and IDC)*, pages 44–51, 2009.

[63] Lamia Youseff, Maria Butrico, and Dilma Da Silva. Toward a unified ontology of cloud computing. In *2008 Grid Computing Environments Workshop*, pages 1–10. IEEE, 2008.

[64] Antonio Regalado. Who coined cloud computing. *Technology Review*, 31, 2011.

[65] Ling Qian, Zhiguo Luo, Yujian Du, and Leitao Guo. Cloud computing: an overview. In *IEEE International Conference on Cloud Computing*, pages 626–631. Springer, 2009.

[66] Sindhu Srivastava, Vani Trehan, Priyanka Yadav, Neha Manga, and Sakshi Gupta. Google app engine. *International Journal of Engineering and Innovative Technology (IJEIT) Volume*, 2.

[67] Nabil Sultan. Cloud computing for education: A new dawn? *International Journal of Information Management*, 30(2):109–116, 2010.

[68] Lizhe Wang, Gregor Von Laszewski, Andrew Younge, Xi He, Marcel Kunze, Jie Tao, and Cheng Fu. Cloud computing: a perspective study. *New Generation Computing*, 28(2):137–146, 2010.

[69] Sarah Gordon and Richard Ford. On the definition and classification of cybercrime. *Journal in Computer Virology*, 2(1):13–20, 2006.

[70] Niels Provos, Moheeb Abu Rajab, and Panayiotis Mavrommatis. Cybercrime 2.0: when the cloud turns dark. *Communications of the ACM*, 52(4):42–47, 2009.

[71] Ross Anderson, Chris Barton, Rainer Böhme, Richard Clayton, Michel JG Van Eeten, Michael Levi, Tyler Moore, and Stefan Savage. Measuring the cost of cybercrime. In *The economics of information security and privacy*, pages 265–300. Springer, 2013.

[72] David L Carter. Computer crime categories: how techno-criminals operate. *FBI Law Enforcement Bulletin*, 64(7):21, 1995.

[73] Marc D Goodman. Why the police don't care about computer crime. *Harv. JL & Tech.*, 10:465, 1996.

[74] Melanie Kowalski. *Cyber-crime: Issues, data sources, and feasibility of collecting police-reported statistics*. Canadian Centre for Justice Statistics, 2002.

[75] Kleber Vieira, Alexandre Schulter, Carlos Westphall, and Carla Merkle Westphall. Intrusion detection for grid and cloud computing. *IT Professional*, 12(4):38–43, 2010.

[76] Wayne A Jansen. Cloud hooks: Security and privacy issues in cloud computing. In *System Sciences (HICSS), 2011 44th Hawaii International Conference on*, pages 1–10. IEEE, 2011.

[77] Jon Oberheide, Evan Cooke, and Farnam Jahanian. Empirical exploitation of live virtual machine migration. In *Proc. of BlackHat DC Convention*. Citeseer, 2008.

[78] Jinpeng Wei, Xiaolan Zhang, Glenn Ammons, Vasanth Bala, and Peng Ning. Managing security of virtual machine images in a cloud environment. In *Proceedings of the 2009 ACM Workshop on Cloud Computing Security*, pages 91–96. ACM, 2009.

[79] Neal Leavitt. Is cloud computing really ready for prime time. *Growth*, 27(5):15–20, 2009.

[80] Chirag Modi, Dhiren Patel, Bhavesh Borisaniya, Avi Patel, and Muttukrishnan Rajarajan. A survey on security issues and solutions at different layers of cloud computing. *The Journal of Supercomputing*, 63(2):561–592, 2013.

[81] BI aya Chandrareddy, G Uma Mahesh, and Sreekanth Bandi. Cloud zones: security and privacy issues in cloud computing. *Asian Journal of Information Technology*, 11(3):83–93, 2012.

[82] Khalid Alhamazani, Rajiv Ranjan, Karan Mitra, Fethi Rabhi, Prem Prakash Jayaraman, Samee Ullah Khan, Adnene Guabtni, and Vasudha Bhatnagar. An overview of the commercial cloud monitoring tools: research dimensions, design issues, and state-of-the-art. *Computing*, 97(4):357–377, 2015.

[83] Jin Shao, Hao Wei, Qianxiang Wang, and Hong Mei. A runtime model based monitoring approach for cloud. In *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, pages 313–320. IEEE, 2010.

[84] Kenneth P Birman. *Guide to Reliable Distributed Systems: Building High-Assurance Applications and Cloud-Hosted Services*. Springer-Verlag New York Incorporated, 2012.

[85] Smruti Padhy, Diego Kreutz, António Casimiro, and Marcelo Pasin. Trustworthy and resilient monitoring system for cloud infrastructures. In *Proceedings of the Workshop on Posters and Demos Track*, page 3. ACM, 2011.

[86] SIA Zabbix. Zabbix. *The Enterprise-class Monitoring Solution for Everyone. www.zabbix.com*, 2015.

[87] Rajkumar Buyya, Chee Shin Yeo, and Srikumar Venugopal. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In *High Performance Computing and Communications, 2008. HPCC'08. 10th IEEE International Conference on*, pages 5–13. Ieee, 2008.

[88] Qian Wang, Cong Wang, Jin Li, Kui Ren, and Wenjing Lou. Enabling public verifiability and data dynamics for storage security in cloud computing. In *European symposium on research in computer security*, pages 355–370. Springer, 2009.

[89] Sara Qaisar and Kausar Fiaz Khawaja. Cloud computing: network/security threats and countermeasures. *Interdisciplinary Journal of Contemporary Research in Business*, 3(9):1323, 2012.

[90] Mohand Mezmaz, Nouredine Melab, Yacine Kessaci, Young Choon Lee, E-G Talbi, Albert Y Zomaya, and Daniel Tuyttens. A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems. *Journal of Parallel and Distributed Computing*, 71(11):1497–1508, 2011.

[91] Rong Ge, Xizhou Feng, and Kirk W Cameron. Performance-constrained distributed dvs scheduling for scientific applications on power-aware clusters. In *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, page 34. IEEE Computer Society, 2005.

[92] Dakai Zhu, Rami Melhem, and Bruce R Childers. Scheduling with dynamic voltage/speed adjustment using slack reclamation in multiprocessor real-time systems. *IEEE Transactions on Parallel and Distributed Systems*, 14(7):686–700, 2003.

[93] Kyong Hoon Kim, Rajkumar Buyya, and Jong Kim. Power aware scheduling of bag-of-tasks applications with deadline constraints on dvs-enabled clusters. In *CCGRID*, volume 7, pages 541–548, 2007.

[94] Jian-Jia Chen and Tei-Wei Kuo. Multiprocessor energy-efficient scheduling for real-time tasks with different power characteristics. In *2005 International Conference on Parallel Processing (ICPP'05)*, pages 13–20. IEEE, 2005.

[95] Barry Rountree, David K Lowenthal, Shelby Funk, Vincent W Freeh, Bronis R De Supinski, and Martin Schulz. Bounding energy consumption in large-scale mpi programs. In *Proceedings of the 2007 ACM/IEEE conference on Supercomputing*, page 49. ACM, 2007.

[96] Zhiguo Wan, Jun'e Liu, and Robert H Deng. Hasbe: a hierarchical attribute-based solution for flexible and scalable access control in cloud computing. *IEEE transactions on information forensics and security*, 7(2):743–754, 2012.

[97] Matthew L Massie, Brent N Chun, and David E Culler. The ganglia distributed monitoring system: design, implementation, and experience. *Parallel Computing*, 30(7):817–840, 2004.

[98] Rodrigo N Calheiros, Rajiv Ranjan, Anton Beloglazov, César AF De Rose, and Rajkumar Buyya. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience*, 41(1):23–50, 2011.

[99] Keith Harrison. Virtual machines, September 30 2010. US Patent App. 13/822,239.

[100] Zhang Xu, Haining Wang, Zichen Xu, and Xiaorui Wang. Power attack: An increasing threat to data centers. In *The Network and Distributed System Security Symposium (NDSS)*, 2014.

[101] Jakub Szefer, Pramod Jamkhedkar, Diego Perez-Botero, and Ruby B Lee. Cyber defenses for physical attacks and insider threats in cloud computing. In *Proceedings of the 9th ACM symposium on Information, computer and communications security*, pages 519–524. ACM, 2014.

[102] Hang Yuan, C-C Jay Kuo, and Ishfaq Ahmad. Energy efficiency in data centers and cloud-based multimedia services: An overview and future directions. In *Green Computing Conference, 2010 International*, pages 375–382. IEEE, 2010.

[103] H Wey. Über die gleichverteilung von zahlen modulo eins. *Mathematische Annalen*, 77:313–352, 1916.

[104] K Athreya, D McDonald, and P Ney. Coupling and the renewal theorem. *The American Mathematical Monthly*, 85(10):809–814, 1978.

[105] David H Arnold. The mécanique physique of siméon denis poisson: The evolution and isolation in france of his approach to physical theory (1800–1840). *Archive for history of exact sciences*, 28(3):243–266, 1983.

[106] Frank A Haight. Handbook of the poisson distribution. Technical report, 1967.

[107] Yvan Royon, Stéphane Frénot, and Frédéric Le Mouël. Virtualization of service gateways in multi-provider environments. In *Component-Based Software Engineering*, pages 385–392. Springer, 2006.

[108] Hans Löhr, Ahmad-Reza Sadeghi, and Marcel Winandy. Securing the e-health cloud. In *Proceedings of the 1st ACM International Health Informatics Symposium*, pages 220–229. ACM, 2010.

[109] Xuehai Zhang, Katarzyna Keahey, Ian Foster, and Timothy Freeman. Virtual cluster workspaces for grid applications. Technical report, Argonne National Laboratory, University of Chicago, 2005.

[110] Kaleem Ullah and MNA Khan. Security and privacy issues in cloud computing environment: A survey paper. *International Journal of Grid and Distributed Computing*, 7(2):89–98, 2014.

[111] Karthik Mahendra Raje Urs. Harnessing the cloud for securely outsourcing large-scale systems of linear equations. *IEEE Transactions on Parallel and Distributed Systems 24.6: 1172-1181.*, 2013.

[112] Fu Xie and Fangai Liu. Dynamic effective resource allocation based on cloud computing learning model. *Journal of Networks*, 9(11):3092–3097, 2014.

[113] Ronald G Minnich and Don W Rudish. Ten million and one penguins, or, lessons learned from booting millions of virtual machines on hpc systems. In *Workshop on System-level Virtualization for High Performance Computing in conjunction with EuroSys*, volume 10. Citeseer, 2009.

[114] Adel Nadjaran Toosi, Rodrigo N Calheiros, Ruppa K Thulasiram, and Rajkumar Buyya. Resource provisioning policies to increase iaas provider's profit in a federated cloud environment. In *High Performance Computing and Communications (HPCC), 2011 IEEE 13th International Conference on*, pages 279–287. IEEE, 2011.

# Appendix A

# Data Sets

This first appendix chapter includes all next tables that include the exact values obtained from experiments and depicted graphically in figures throughout the whole thesis.

Tables A.1, A.2 and A.3 present data that is used in Figures 5.1, 5.6, 5.7, 5.8 and 5.3. The columns present each data set. However, the first column present the number of VMs.

Tables A.4, A.5, A.6, A.7, A.8, A.9, A.10, A.11, A.12 and A.13 show data that is used for the two versions of reduction. The columns in all tables present the highest ranges.

Finally, the rest of tables present data about symptoms appearance, hierarchal structure and balancing Mobility Algorithm. The columns in all tables present the highest ranges. However, the columns in hierarchal structure tables present the clusters results. The red values in all tables present the best result of the highest ranges for Randomised-Local Reduction.

Table A.1: Reduce-Max All data Part 1

| No. of VMs | Arithmetic | Uniform | Harmonic | Exponential | Radom |
|---|---|---|---|---|---|
| 5 | 1.3333 | 1.0000 | 1.3139 | 1.0323 | 1.2632 |
| 10 | 1.4182 | 1.0000 | 1.3657 | 1.0010 | 1.4651 |
| 15 | 1.4167 | 1.0000 | 1.5068 | 1.0000 | 1.2941 |
| 20 | 1.4286 | 1.0000 | 1.3898 | 1.0000 | 1.2308 |
| 25 | 1.4123 | 1.0000 | 1.3103 | 1.0000 | 1.1855 |
| 30 | 1.4194 | 1.0000 | 1.3141 | 1.0000 | 1.2895 |
| 35 | 1.4238 | 1.0000 | 1.3263 | 1.0000 | 1.2426 |
| 40 | 1.3951 | 1.0000 | 1.4023 | 1.0000 | 1.2000 |
| 45 | 1.3913 | 1.0000 | 1.3652 | 1.0000 | 1.2245 |
| 50 | 1.3867 | 1.0000 | 1.3336 | 1.0000 | 1.1462 |
| 55 | 1.3935 | 1.0000 | 1.3062 | 1.0000 | 1.1765 |
| 60 | 1.4098 | 1.0000 | 1.2821 | 1.0000 | 1.1985 |
| 65 | 1.3636 | 1.0000 | 1.2607 | 1.0000 | 1.1699 |
| 70 | 1.3907 | 1.0000 | 1.2415 | 1.0000 | 1.1483 |
| 75 | 1.3684 | 1.0000 | 1.2922 | 1.0000 | 1.2059 |
| 80 | 1.3741 | 1.0000 | 1.2755 | 1.0000 | 1.2070 |
| 85 | 1.3488 | 1.0000 | 1.2602 | 1.0000 | 1.1657 |
| 90 | 1.3407 | 1.0000 | 1.2461 | 1.0000 | 1.1735 |
| 95 | 1.3542 | 1.0000 | 1.2330 | 1.0000 | 1.2200 |
| 100 | 1.3515 | 1.0000 | 1.2209 | 1.0000 | 1.2202 |
| 105 | 1.3441 | 1.0000 | 1.2096 | 1.0000 | 1.1739 |
| 110 | 1.3392 | 1.0000 | 1.1990 | 1.0000 | 1.1265 |
| 115 | 1.3793 | 1.0000 | 1.1890 | 1.0000 | 1.1904 |
| 120 | 1.3388 | 1.0000 | 1.3038 | 1.0000 | 1.1587 |
| 125 | 1.3432 | 1.0000 | 1.2940 | 1.0000 | 1.2031 |
| 130 | 1.3332 | 1.0000 | 1.2847 | 1.0000 | 1.1475 |
| 135 | 1.3325 | 1.0000 | 1.2759 | 1.0000 | 1.1818 |
| 140 | 1.3617 | 1.0000 | 1.2676 | 1.0000 | 1.1856 |
| 145 | 1.3288 | 1.0000 | 1.2596 | 1.0000 | 1.1648 |
| 150 | 1.3642 | 1.0000 | 1.2520 | 1.0000 | 1.1724 |
| 155 | 1.3246 | 1.0000 | 1.2447 | 1.0000 | 1.1926 |
| 160 | 1.3242 | 1.0000 | 1.2377 | 1.0000 | 1.1602 |
| 165 | 1.3170 | 1.0000 | 1.2311 | 1.0000 | 1.2017 |
| 170 | 1.3216 | 1.0000 | 1.2246 | 1.0000 | 1.1813 |
| 175 | 1.3295 | 1.0000 | 1.2185 | 1.0000 | 1.1771 |
| 180 | 1.3260 | 1.0000 | 1.2126 | 1.0000 | 1.1871 |
| 185 | 1.3369 | 1.0000 | 1.2068 | 1.0000 | 1.1550 |
| 190 | 1.3227 | 1.0000 | 1.2013 | 1.0000 | 1.1597 |
| 195 | 1.3367 | 1.0000 | 1.1960 | 1.0000 | 1.1749 |
| 200 | 1.3134 | 1.0000 | 1.1909 | 1.0000 | 1.1813 |

Table A.2: Reduce-Max All data Part 2

| No. of VMs | Arithmetic | Uniform | Harmonic | Exponential | Radom |
|---|---|---|---|---|---|
| 205 | 1.3301 | 1.0000 | 1.1859 | 1.0000 | 1.1905 |
| 210 | 1.3270 | 1.0000 | 1.1811 | 1.0000 | 1.1779 |
| 215 | 1.3264 | 1.0000 | 1.2325 | 1.0000 | 1.1504 |
| 220 | 1.3176 | 1.0000 | 1.1719 | 1.0000 | 1.1635 |
| 225 | 1.3274 | 1.0000 | 1.3343 | 1.0000 | 1.1971 |
| 230 | 1.3160 | 1.0000 | 1.2187 | 1.0000 | 1.1552 |
| 235 | 1.3076 | 1.0000 | 1.1591 | 1.0000 | 1.1673 |
| 240 | 1.3195 | 1.0000 | 1.3201 | 1.0000 | 1.1670 |
| 245 | 1.3108 | 1.0000 | 1.2060 | 1.0000 | 1.1721 |
| 250 | 1.3157 | 1.0000 | 1.2021 | 1.0000 | 1.1642 |
| 255 | 1.3070 | 1.0000 | 1.1982 | 1.0000 | 1.1613 |
| 260 | 1.3104 | 1.0000 | 1.1944 | 1.0000 | 1.1802 |
| 265 | 1.3225 | 1.0000 | 1.2178 | 1.0000 | 1.1690 |
| 270 | 1.3127 | 1.0000 | 1.2950 | 1.0000 | 1.1702 |
| 275 | 1.3065 | 1.0000 | 1.1836 | 1.0000 | 1.1617 |
| 280 | 1.3238 | 1.0000 | 1.1802 | 1.0000 | 1.2027 |
| 285 | 1.3147 | 1.0000 | 1.1768 | 1.0000 | 1.1906 |
| 290 | 1.3127 | 1.0000 | 1.1736 | 1.0000 | 1.1653 |
| 295 | 1.3099 | 1.0000 | 1.2768 | 1.0000 | 1.1965 |
| 300 | 1.3148 | 1.0000 | 1.1672 | 1.0000 | 1.1870 |
| 305 | 1.3062 | 1.0000 | 1.1642 | 1.0000 | 1.1603 |
| 310 | 1.3055 | 1.0000 | 1.1612 | 1.0000 | 1.1779 |
| 315 | 1.3139 | 1.0000 | 1.1583 | 1.0000 | 1.1593 |
| 320 | 1.3048 | 1.0000 | 1.1554 | 1.0000 | 1.1706 |
| 325 | 1.3190 | 1.0000 | 1.1526 | 1.0000 | 1.1689 |
| 330 | 1.3081 | 1.0000 | 1.1759 | 1.0000 | 1.1934 |
| 335 | 1.3036 | 1.0000 | 1.1471 | 1.0000 | 1.1871 |
| 340 | 1.3150 | 1.0000 | 1.2485 | 1.0000 | 1.1774 |
| 345 | 1.3040 | 1.0000 | 1.2457 | 1.0000 | 1.1644 |
| 350 | 1.3048 | 1.0000 | 1.2429 | 1.0000 | 1.1768 |
| 355 | 1.3146 | 1.0000 | 1.1885 | 1.0000 | 1.1742 |
| 360 | 1.3019 | 1.0000 | 1.1601 | 1.0000 | 1.1728 |
| 365 | 1.3097 | 1.0000 | 1.2349 | 1.0000 | 1.1610 |
| 370 | 1.3315 | 1.0000 | 1.1399 | 1.0000 | 1.1653 |
| 375 | 1.3093 | 1.0000 | 1.2297 | 1.0000 | 1.1784 |
| 380 | 1.3123 | 1.0000 | 1.2272 | 1.0000 | 1.1753 |
| 385 | 1.3122 | 1.0000 | 1.2248 | 1.0000 | 1.1824 |
| 390 | 1.3110 | 1.0000 | 1.2224 | 1.0000 | 1.1803 |
| 395 | 1.3009 | 1.0000 | 1.2200 | 1.0000 | 1.1754 |
| 400 | 1.3017 | 1.0000 | 1.2177 | 1.0000 | 1.1754 |

Table A.3: Reduce-Max All data Part 3

| No. of VMs | Arithmetic | Uniform | Harmonic | Exponential | Radom |
|---|---|---|---|---|---|
| 405 | 1.3016 | 1.0000 | 1.2154 | 1.0000 | 1.1681 |
| 410 | 1.3187 | 1.0000 | 1.2131 | 1.0000 | 1.1693 |
| 415 | 1.3077 | 1.0000 | 1.2109 | 1.0000 | 1.1786 |
| 420 | 1.3159 | 1.0000 | 1.2087 | 1.0000 | 1.1625 |
| 425 | 1.3099 | 1.0000 | 1.2065 | 1.0000 | 1.1697 |
| 430 | 1.2960 | 1.0000 | 1.2044 | 1.0000 | 1.1982 |
| 435 | 1.3028 | 1.0000 | 1.2023 | 1.0000 | 1.1661 |
| 440 | 1.3164 | 1.0000 | 1.2003 | 1.0000 | 1.1707 |
| 445 | 1.3049 | 1.0000 | 1.1982 | 1.0000 | 1.1949 |
| 450 | 1.3065 | 1.0000 | 1.1962 | 1.0000 | 1.1606 |
| 455 | 1.3125 | 1.0000 | 1.1943 | 1.0000 | 1.1888 |
| 460 | 1.3115 | 1.0000 | 1.1923 | 1.0000 | 1.1653 |
| 465 | 1.2931 | 1.0000 | 1.1904 | 1.0000 | 1.1650 |
| 470 | 1.2964 | 1.0000 | 1.1885 | 1.0000 | 1.1873 |
| 475 | 1.2941 | 1.0000 | 1.1867 | 1.0000 | 1.1822 |
| 480 | 1.3015 | 1.0000 | 1.1848 | 1.0000 | 1.1921 |
| 485 | 1.3169 | 1.0000 | 1.1830 | 1.0000 | 1.1742 |
| 490 | 1.3035 | 1.0000 | 1.1812 | 1.0000 | 1.1725 |
| 495 | 1.3086 | 1.0000 | 1.1795 | 1.0000 | 1.1735 |
| 500 | 1.3253 | 1.0000 | 1.1777 | 1.0000 | 1.1801 |
| 505 | 1.3004 | 1.0000 | 1.1760 | 1.0000 | 1.1758 |
| 510 | 1.3114 | 1.0000 | 1.1743 | 1.0000 | 1.1807 |
| 515 | 1.3062 | 1.0000 | 1.1726 | 1.0000 | 1.1711 |
| 520 | 1.2821 | 1.0000 | 1.1710 | 1.0000 | 1.1624 |
| 525 | 1.2814 | 1.0000 | 1.1693 | 1.0000 | 1.1637 |
| 530 | 1.2858 | 1.0000 | 1.1677 | 1.0000 | 1.1642 |
| 535 | 1.2873 | 1.0000 | 1.1661 | 1.0000 | 1.1687 |
| 540 | 1.2754 | 1.0000 | 1.1645 | 1.0000 | 1.1647 |
| 545 | 1.3071 | 1.0000 | 1.1630 | 1.0000 | 1.1716 |
| 550 | 1.3176 | 1.0000 | 1.1614 | 1.0000 | 1.1673 |
| 555 | 1.3058 | 1.0000 | 1.1599 | 1.0000 | 1.1701 |
| 560 | 1.2941 | 1.0000 | 1.1584 | 1.0000 | 1.1807 |
| 565 | 1.3145 | 1.0000 | 1.1569 | 1.0000 | 1.1645 |
| 570 | 1.2925 | 1.0000 | 1.1554 | 1.0000 | 1.1660 |
| 575 | 1.2951 | 1.0000 | 1.2021 | 1.0000 | 1.1642 |
| 580 | 1.3195 | 1.0000 | 1.2006 | 1.0000 | 1.1593 |
| 585 | 1.2935 | 1.0000 | 1.1511 | 1.0000 | 1.1688 |
| 590 | 1.2883 | 1.0000 | 1.1497 | 1.0000 | 1.1684 |
| 595 | 1.3197 | 1.0000 | 1.1483 | 1.0000 | 1.1525 |
| 600 | 1.3178 | 1.0000 | 1.1470 | 1.0000 | 1.1586 |

Table A.4: The two versions of reduction for Uniform weights Part 1

| Highest | det. | 8 h | 7 h | 6 h | 5 h | 4 h | 3 h | 2 h |
|---|---|---|---|---|---|---|---|---|
| 1 FVM | 1.00 | 1.10 | 1.08 | 1.06 | 1.05 | 1.04 | 1.03 | 1.02 |
| 2 FVMs | 0.50 | 0.58 | 0.58 | 0.59 | 0.59 | 0.60 | 0.63 | 0.70 |
| 3 FVMs | 0.33 | 0.42 | 0.42 | 0.42 | 0.44 | 0.46 | 0.50 | 0.60 |
| 4 FVMs | 0.25 | 0.33 | 0.34 | 0.34 | 0.36 | 0.39 | 0.44 | 0.55 |
| 5 FVMs | 0.20 | 0.28 | 0.29 | 0.30 | 0.32 | 0.34 | 0.40 | 0.53 |
| 6 FVMs | 0.17 | 0.25 | 0.26 | 0.27 | 0.29 | 0.32 | 0.38 | 0.52 |
| 7 FVMs | 0.14 | 0.22 | 0.23 | 0.25 | 0.27 | 0.30 | 0.37 | 0.51 |
| 8 FVMs | 0.13 | 0.21 | 0.22 | 0.23 | 0.25 | 0.29 | 0.36 | 0.51 |
| 9 FVMs | 0.11 | 0.19 | 0.20 | 0.22 | 0.24 | 0.28 | 0.35 | 0.50 |
| 10 FVMs | 0.10 | 0.18 | 0.19 | 0.21 | 0.24 | 0.27 | 0.35 | 0.50 |
| 11 FVMs | 0.09 | 0.17 | 0.19 | 0.20 | 0.23 | 0.27 | 0.34 | 0.50 |
| 12 FVMs | 0.08 | 0.17 | 0.18 | 0.20 | 0.22 | 0.27 | 0.34 | 0.50 |
| 13 FVMs | 0.08 | 0.16 | 0.18 | 0.19 | 0.22 | 0.26 | 0.34 | 0.50 |
| 14 FVMs | 0.07 | 0.16 | 0.17 | 0.19 | 0.22 | 0.26 | 0.34 | 0.50 |
| 15 FVMs | 0.07 | 0.15 | 0.17 | 0.19 | 0.21 | 0.26 | 0.34 | 0.50 |
| 16 FVMs | 0.06 | 0.15 | 0.16 | 0.18 | 0.21 | 0.26 | 0.34 | 0.50 |

Table A.5: The two versions of reduction for Uniform weights Part 2

| Highest | 16 h | 15 h | 14 h | 13 h | 12 h | 11 h | 10 h | 9 h |
|---|---|---|---|---|---|---|---|---|
| 1 FVM | 1.17 | 1.20 | 1.16 | 1.14 | 1.15 | 1.14 | 1.12 | 1.09 |
| 2 FVMs | 0.63 | 0.61 | 0.61 | 0.61 | 0.60 | 0.59 | 0.58 | 0.58 |
| 3 FVMs | 0.44 | 0.42 | 0.43 | 0.42 | 0.41 | 0.41 | 0.41 | 0.41 |
| 4 FVMs | 0.33 | 0.32 | 0.32 | 0.32 | 0.32 | 0.32 | 0.32 | 0.33 |
| 5 FVMs | 0.27 | 0.27 | 0.27 | 0.27 | 0.27 | 0.27 | 0.27 | 0.27 |
| 6 FVMs | 0.23 | 0.23 | 0.23 | 0.23 | 0.23 | 0.23 | 0.24 | 0.24 |
| 7 FVMs | 0.20 | 0.20 | 0.20 | 0.20 | 0.21 | 0.21 | 0.21 | 0.22 |
| 8 FVMs | 0.18 | 0.18 | 0.18 | 0.19 | 0.19 | 0.19 | 0.19 | 0.20 |
| 9 FVMs | 0.17 | 0.17 | 0.17 | 0.17 | 0.17 | 0.17 | 0.18 | 0.19 |
| 10 FVMs | 0.15 | 0.15 | 0.16 | 0.16 | 0.16 | 0.16 | 0.17 | 0.17 |
| 11 FVMs | 0.14 | 0.15 | 0.15 | 0.15 | 0.15 | 0.15 | 0.16 | 0.17 |
| 12 FVMs | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.15 | 0.15 | 0.16 |
| 13 FVMs | 0.13 | 0.13 | 0.13 | 0.14 | 0.14 | 0.14 | 0.15 | 0.15 |
| 14 FVMs | 0.12 | 0.12 | 0.13 | 0.13 | 0.13 | 0.14 | 0.14 | 0.15 |
| 15 FVMs | 0.12 | 0.12 | 0.12 | 0.12 | 0.13 | 0.13 | 0.14 | 0.14 |
| 16 FVMs | 0.11 | 0.11 | 0.12 | 0.12 | 0.12 | 0.13 | 0.13 | 0.14 |

Table A.6: The two versions of reduction for Arithmetic weights Part 1

| FVMs | deter. | 9 h | 8 h | 7 h | 6 h | 5 h | 4 h | 3 h | 2 h |
|---|---|---|---|---|---|---|---|---|---|
| 16 | 0.0688 | 0.1470 | 0.1554 | 0.1705 | 0.1994 | 0.218 | 0.2680 | 0.3468 | 0.5125 |
| 15 | 0.0741 | 0.1487 | 0.1575 | 0.1720 | 0.2012 | 0.2207 | 0.2681 | 0.3473 | 0.5121 |
| 14 | 0.0787 | 0.1527 | 0.1616 | 0.1765 | 0.2035 | 0.2223 | 0.2697 | 0.3468 | 0.5118 |
| 13 | 0.0858 | 0.1592 | 0.1677 | 0.1799 | 0.2056 | 0.2273 | 0.2726 | 0.3478 | 0.5133 |
| 12 | 0.0925 | 0.1666 | 0.1741 | 0.1841 | 0.2118 | 0.2299 | 0.2730 | 0.3491 | 0.5123 |
| 11 | 0.1014 | 0.1736 | 0.1797 | 0.1942 | 0.2066 | 0.2343 | 0.2775 | 0.3503 | 0.5132 |
| 10 | 0.1056 | 0.1818 | 0.1885 | 0.1992 | 0.2146 | 0.2438 | 0.2806 | 0.3534 | 0.5132 |
| 9 | 0.1173 | 0.1948 | 0.2008 | 0.2090 | 0.2232 | 0.2470 | 0.2864 | 0.3569 | 0.5138 |
| 8 | 0.1346 | 0.2077 | 0.2158 | 0.2221 | 0.2364 | 0.2592 | 0.2938 | 0.3625 | 0.5145 |
| 7 | 0.1588 | 0.2265 | 0.2383 | 0.2466 | 0.2532 | 0.2714 | 0.3050 | 0.3696 | 0.5175 |
| 6 | 0.1878 | 0.2514 | 0.2582 | 0.2622 | 0.2764 | 0.2928 | 0.3238 | 0.3834 | 0.5218 |
| 5 | 0.2127 | 0.2892 | 0.2937 | 0.2973 | 0.3063 | 0.3225 | 0.3496 | 0.4033 | 0.5328 |
| 4 | 0.2644 | 0.3499 | 0.3436 | 0.3464 | 0.3554 | 0.3673 | 0.3902 | 0.4389 | 0.5522 |
| 3 | 0.3457 | 0.4410 | 0.4421 | 0.4347 | 0.4420 | 0.4466 | 0.4628 | 0.5014 | 0.5966 |
| 2 | 0.5122 | 0.6209 | 0.6134 | 0.6175 | 0.6063 | 0.6036 | 0.6134 | 0.6359 | 0.7009 |
| 1 | 1.0122 | 1.1854 | 1.1637 | 1.131 | 1.1125 | 1.0889 | 1.069 | 1.0508 | 1.0368 |

Table A.7: The two versions of reduction for Arithmetic weights Part 2

| FVMs | 16 h | 15 h | 14 h | 13 h | 12 h | 11 h | 10 h |
|---|---|---|---|---|---|---|---|
| 1 | 1.3263 | 1.2913 | 1.3494 | 1.2519 | 1.2395 | 1.1961 | 1.1883 |
| 2 | 0.6846 | 0.6676 | 0.6593 | 0.6769 | 0.6477 | 0.6379 | 0.6416 |
| 3 | 0.4855 | 0.4610 | 0.4535 | 0.4531 | 0.4412 | 0.4422 | 0.438 |
| 4 | 0.3575 | 0.3655 | 0.3511 | 0.3489 | 0.3459 | 0.3492 | 0.345 |
| 5 | 0.3098 | 0.3030 | 0.2884 | 0.2872 | 0.2858 | 0.2864 | 0.2914 |
| 6 | 0.2669 | 0.2531 | 0.2453 | 0.2543 | 0.2517 | 0.2475 | 0.2528 |
| 7 | 0.2247 | 0.2254 | 0.2278 | 0.2260 | 0.2203 | 0.2252 | 0.2251 |
| 8 | 0.2020 | 0.2054 | 0.2085 | 0.2035 | 0.2008 | 0.2023 | 0.2051 |
| 9 | 0.1817 | 0.1874 | 0.1846 | 0.1796 | 0.1864 | 0.1901 | 0.1948 |
| 10 | 0.1726 | 0.1688 | 0.1719 | 0.1686 | 0.1719 | 0.1750 | 0.1784 |
| 11 | 0.1586 | 0.1578 | 0.1579 | 0.1650 | 0.1609 | 0.1641 | 0.1678 |
| 12 | 0.1543 | 0.1470 | 0.1505 | 0.1535 | 0.1503 | 0.1553 | 0.1614 |
| 13 | 0.1387 | 0.1401 | 0.1408 | 0.1433 | 0.1433 | 0.1473 | 0.1570 |
| 14 | 0.1315 | 0.1334 | 0.1341 | 0.1342 | 0.1385 | 0.1414 | 0.1468 |
| 15 | 0.1285 | 0.1281 | 0.1321 | 0.1335 | 0.1336 | 0.1369 | 0.1418 |
| 16 | 0.1202 | 0.1275 | 0.1240 | 0.1302 | 0.1294 | 0.1353 | 0.1385 |

Table A.8: The two versions of reduction for Random weights Part 1

| FVMs | deter. | 8 h | 7 h | 6 h | 5 h | 4 h | 3 h | 2 h |
|---|---|---|---|---|---|---|---|---|
| 16 | 0.0751 | 0.1720 | 0.1873 | 0.208549 | 0.245404 | 0.298759 | 0.392519 | 0.5859 |
| 15 | 0.0790 | 0.1745 | 0.1886 | 0.214286 | 0.245738 | 0.30062 | 0.394336 | 0.5936 |
| 14 | 0.0844 | 0.1780 | 0.1925 | 0.216651 | 0.25052 | 0.301662 | 0.39518 | 0.5902 |
| 13 | 0.0908 | 0.1846 | 0.1984 | 0.219553 | 0.250389 | 0.304624 | 0.395378 | 0.5923 |
| 12 | 0.0983 | 0.1941 | 0.2045 | 0.225447 | 0.254828 | 0.308273 | 0.394836 | 0.5917 |
| 11 | 0.1073 | 0.198 | 0.2115 | 0.231472 | 0.262319 | 0.311653 | 0.397041 | 0.5943 |
| 10 | 0.1182 | 0.2090 | 0.2197 | 0.238538 | 0.267761 | 0.315763 | 0.402256 | 0.5888 |
| 9 | 0.1309 | 0.2188 | 0.2315 | 0.249749 | 0.274949 | 0.322552 | 0.409834 | 0.5886 |
| 8 | 0.1472 | 0.2323 | 0.2433 | 0.264877 | 0.288749 | 0.331431 | 0.41237 | 0.593 |
| 7 | 0.1675 | 0.2520 | 0.2637 | 0.279221 | 0.301588 | 0.344643 | 0.419825 | 0.5942 |
| 6 | 0.1959 | 0.2802 | 0.2931 | 0.306195 | 0.325791 | 0.366552 | 0.435088 | 0.6007 |
| 5 | 0.2352 | 0.3165 | 0.3242 | 0.338292 | 0.358026 | 0.391685 | 0.457635 | 0.6131 |
| 4 | 0.292 | 0.3713 | 0.3800 | 0.388889 | 0.409189 | 0.440379 | 0.493246 | 0.6344 |
| 3 | 0.390 | 0.4656 | 0.4807 | 0.48444 | 0.49453 | 0.532998 | 0.573009 | 0.6816 |
| 2 | 0.5860 | 0.6806 | 0.6681 | 0.669104 | 0.667969 | 0.685934 | 0.714836 | 0.7990 |
| 1 | 1.1689 | 1.2543 | 1.2352 | 1.23285 | 1.217487 | 1.210358 | 1.191802 | 1.1873 |

Table A.9: The two versions of reduction for Random weights Part 2

| FVMs | 16 h | 15 h | 14 h | 13 h | 12 h | 11 h | 10 h | 9 h |
|---|---|---|---|---|---|---|---|---|
| 16 | 0.1313 | 0.128792 | 0.12799 | 0.1343 | 0.136686 | 0.142514 | 0.1521 | 0.1588 |
| 15 | 0.1290 | 0.131767 | 0.1322 | 0.139 | 0.141829 | 0.150012 | 0.154 | 0.164 |
| 14 | 0.1335 | 0.137367 | 0.1397 | 0.144446 | 0.1474 | 0.155494 | 0.1610 | 0.167 |
| 13 | 0.1453 | 0.143913 | 0.14977 | 0.1527 | 0.152025 | 0.159318 | 0.1669 | 0.1736 |
| 12 | 0.1516 | 0.152428 | 0.1554 | 0.1558 | 0.16106 | 0.168743 | 0.1721 | 0.1806 |
| 11 | 0.1617 | 0.160055 | 0.165 | 0.1684 | 0.173472 | 0.173914 | 0.1791 | 0.1898 |
| 10 | 0.1793 | 0.173794 | 0.1769 | 0.1810 | 0.181041 | 0.188008 | 0.1889 | 0.1962 |
| 9 | 0.1860 | 0.188447 | 0.188306 | 0.191 | 0.197143 | 0.200598 | 0.2027 | 0.2092 |
| 8 | 0.2067 | 0.208753 | 0.211465 | 0.2033 | 0.210884 | 0.214022 | 0.2180 | 0.2244 |
| 7 | 0.2264 | 0.228843 | 0.228862 | 0.2279 | 0.232416 | 0.232473 | 0.2415 | 0.2410 |
| 6 | 0.2627 | 0.263564 | 0.258376 | 0.2620 | 0.26214 | 0.261 | 0.2644 | 0.2699 |
| 5 | 0.3093 | 0.301448 | 0.301535 | 0.3058 | 0.302076 | 0.297564 | 0.3066 | 0.3151 |
| 4 | 0.3655 | 0.374883 | 0.368171 | 0.3644 | 0.360394 | 0.364106 | 0.371 | 0.3672 |
| 3 | 0.4831 | 0.487931 | 0.467745 | 0.4710 | 0.4692 | 0.4716 | 0.4655 | 0.4762 |
| 2 | 0.7101 | 0.695808 | 0.6914 | 0.6892 | 0.674024 | 0.661654 | 0.6809 | 0.6659 |
| 1 | 1.3242 | 1.346185 | 1.380648 | 1.3528 | 1.2852 | 1.274259 | 1.2673 | 1.2608 |

Table A.10: The two versions of reduction for Harmonic weights Part 1

| FVMs | deter. | 8 h | 7 h | 6 h | 5 h | 4 h | 3 h | 2 h |
|---|---|---|---|---|---|---|---|---|
| 16 | 0.1332 | 0.9233 | 0.7901 | 0.7901 | 0.6836 | 0.5327 | 0.5327 | 0.6659 |
| 15 | 0.1332 | 0.9855 | 0.7901 | 0.8700 | 0.6570 | 0.6659 | 0.5416 | 0.6659 |
| 14 | 0.1332 | 1.0654 | 1.0565 | 0.8168 | 0.7280 | 0.6303 | 0.6659 | 0.6659 |
| 13 | 0.1332 | 1.0387 | 0.9588 | 0.9499 | 0.7458 | 0.6481 | 0.6570 | 0.6659 |
| 12 | 0.1332 | 1.1808 | 1.0476 | 0.9322 | 0.9056 | 0.6659 | 0.6659 | 0.6659 |
| 11 | 0.1332 | 1.2784 | 1.1897 | 0.9855 | 0.9943 | 0.6747 | 0.6570 | 0.6659 |
| 10 | 0.1332 | 1.3051 | 1.3317 | 1.1098 | 0.9233 | 0.9144 | 0.7635 | 0.6659 |
| 9 | 0.1332 | 1.5980 | 1.3850 | 1.2696 | 1.0476 | 0.8345 | 0.7990 | 0.6659 |
| 8 | 0.1332 | 1.6957 | 1.5714 | 1.1808 | 1.0476 | 1.0121 | 0.8434 | 0.7458 |
| 7 | 0.2663 | 1.9532 | 1.8555 | 1.4205 | 1.3850 | 1.1630 | 0.9144 | 0.7901 |
| 6 | 0.2663 | 2.4947 | 1.6957 | 1.6069 | 1.3850 | 1.2252 | 0.9499 | 0.8345 |
| 5 | 0.2663 | 2.4858 | 2.4237 | 2.1041 | 1.5803 | 1.4294 | 1.2784 | 0.9233 |
| 4 | 0.3995 | 3.2849 | 3.2760 | 2.4858 | 2.1396 | 1.7312 | 1.3051 | 1.0476 |
| 3 | 0.3995 | 3.8975 | 3.7021 | 3.6045 | 2.4503 | 2.4681 | 1.8289 | 1.4471 |
| 2 | 0.6659 | 5.6464 | 5.0871 | 4.1727 | 3.4979 | 3.1872 | 2.7078 | 1.6957 |
| 1 | 1.1985 | 11.4349 | 11.1863 | 9.1710 | 6.8183 | 6.1436 | 5.0605 | 2.9919 |

Table A.11: The two versions of reduction for Harmonic weights Part 2

| FVMs | 16 h | 15 h | 14 h | 13 h | 12 h | 11 h | 10 h | 9 h |
|---|---|---|---|---|---|---|---|---|
| 16 | 1.7401 | 1.8200 | 1.5714 | 1.3228 | 1.3228 | 1.1985 | 1.4116 | 0.9322 |
| 15 | 1.7135 | 1.8289 | 1.4826 | 1.7667 | 1.3317 | 1.3317 | 1.1808 | 1.0121 |
| 14 | 1.8910 | 1.7578 | 1.8821 | 1.5803 | 1.5714 | 1.4116 | 1.1985 | 1.0210 |
| 13 | 1.8999 | 2.2728 | 1.9088 | 1.6868 | 1.4738 | 1.5714 | 1.2962 | 1.1186 |
| 12 | 2.1485 | 1.9443 | 1.8999 | 1.7934 | 1.6602 | 1.5714 | 1.4649 | 1.4649 |
| 11 | 2.5036 | 2.3971 | 2.1130 | 1.8644 | 1.8466 | 1.8555 | 1.4560 | 1.4560 |
| 10 | 2.9297 | 2.4592 | 2.5302 | 2.1662 | 1.8733 | 2.0508 | 1.8289 | 1.5004 |
| 9 | 3.0452 | 3.0185 | 2.4148 | 2.3527 | 1.9887 | 1.9620 | 1.8378 | 1.7046 |
| 8 | 3.6400 | 3.1162 | 2.8498 | 2.3438 | 2.4592 | 2.3793 | 1.9798 | 1.7578 |
| 7 | 3.4624 | 3.7288 | 3.2937 | 3.0097 | 2.7344 | 2.7078 | 2.0952 | 2.1307 |
| 6 | 3.8442 | 4.0040 | 3.4003 | 3.4713 | 3.3204 | 2.9475 | 2.6101 | 2.3172 |
| 5 | 5.2380 | 4.1904 | 4.5722 | 4.1904 | 3.5068 | 3.4802 | 3.4003 | 2.9741 |
| 4 | 5.8329 | 5.5665 | 5.0516 | 4.8208 | 4.6965 | 4.1283 | 3.8087 | 3.3914 |
| 3 | 7.6884 | 7.4575 | 7.2711 | 5.9660 | 5.5221 | 5.9927 | 4.7320 | 4.9628 |
| 2 | 12.0741 | 10.1476 | 10.9200 | 9.5883 | 8.6561 | 9.1710 | 6.9781 | 6.5431 |
| 1 | 20.3751 | 22.4081 | 19.4784 | 15.2613 | 19.5938 | 13.1128 | 14.9950 | 13.0507 |

Table A.12: The two versions of reduction for Exponential weights Part 1

| FVMs | deter. | 8 h | 7 h | 6 h | 5 h | 4 h | 3 h | 2 h |
|---|---|---|---|---|---|---|---|---|
| 16 | 0.5 | 3.466667 | 3 | 2.6 | 1.966667 | 1.933333 | 1.466667 | 0.966667 |
| 15 | 0.5 | 3.5 | 2.733333 | 2.5 | 1.966667 | 1.9 | 1.466667 | 1 |
| 14 | 0.5 | 3.933333 | 3.5 | 2.5 | 2 | 1.966667 | 1.466667 | 1 |
| 13 | 0.5 | 3.933333 | 3 | 2.866667 | 2.433333 | 1.866667 | 1.5 | 1 |
| 12 | 0.5 | 3.866667 | 3.466667 | 3.5 | 2.5 | 2.1333 | 1.7 | 1 |
| 11 | 0.5 | 4.033333 | 4 | 3.233333 | 2.933333 | 2.4 | 1.9666 | 1.1 |
| 10 | 0.5 | 5 | 5.133333 | 3.866667 | 2.966667 | 2.3667 | 1.8 | 1.1666 |
| 9 | 0.5 | 6.133333 | 4.633333 | 4.433333 | 3.433333 | 2.5667 | 2 | 1.9 |
| 8 | 0.5 | 6.466667 | 5 | 4.633333 | 3.933333 | 2.9333 | 2.4 | 1.5 |
| 7 | 0.5 | 7 | 7.333333 | 4.9 | 4.2 | 3.3 | 2.9333 | 1.8666 |
| 6 | 0.5 | 8.566667 | 7.466667 | 5.5 | 4.566667 | 4.1666 | 2.8666 | 1.8666 |
| 5 | 0.5 | 9.1 | 8.5 | 7.2 | 6.633333 | 5.1333 | 3.4 | 2.0666 |
| 4 | 0.5 | 10.83333 | 10.43333 | 9.333333 | 7.266667 | 5.7333 | 4.3333 | 2.5 |
| 3 | 0.5 | 15.26667 | 13.6 | 11.56667 | 10.4 | 7.6 | 5.4 | 3.7333 |
| 2 | 0.5 | 24.5 | 18.4 | 17.03333 | 15.16667 | 12.6 | 7.6666 | 5.3 |
| 1 | 1 | 45.13333 | 34.03333 | 29.73333 | 27.93333 | 24.5 | 14.433 | 8.5 |

Table A.13: The two versions of reduction for Exponential weights Part 2

| FVMs | 16 h | 15 h | 14 h | 13 h | 12 h | 11 h | 10 h | 9 h |
|---|---|---|---|---|---|---|---|---|
| 16 | 5.5000 | 5.5667 | 5.5333 | 4.9333 | 5.0667 | 4.5667 | 4.0000 | 3.4667 |
| 15 | 6.7333 | 5.8667 | 5.9000 | 5.1667 | 5.0000 | 5.3000 | 5.0000 | 3.8667 |
| 14 | 7.7667 | 6.5000 | 6.5667 | 6.8667 | 5.7000 | 4.9333 | 4.3000 | 4.0000 |
| 13 | 8.0000 | 7.2667 | 7.6667 | 6.9667 | 6.3667 | 5.3667 | 5.2667 | 4.5000 |
| 12 | 8.3333 | 8.3333 | 7.0667 | 7.7000 | 5.5333 | 7.0000 | 5.6333 | 4.2667 |
| 11 | 8.8333 | 8.4333 | 8.1667 | 6.8000 | 6.4667 | 6.7333 | 5.2333 | 4.9333 |
| 10 | 9.9667 | 9.1667 | 8.1000 | 7.5667 | 7.6667 | 6.4667 | 5.8000 | 6.4333 |
| 9 | 11.3333 | 10.8000 | 10.5000 | 10.5000 | 7.7000 | 7.8000 | 6.5000 | 5.2333 |
| 8 | 11.2000 | 10.8667 | 10.0000 | 10.3667 | 8.8333 | 9.5000 | 8.6333 | 6.7000 |
| 7 | 13.1667 | 15.0000 | 12.0000 | 10.8333 | 9.1667 | 10.4333 | 8.2333 | 7.3667 |
| 6 | 14.4333 | 14.9333 | 14.4000 | 12.0000 | 13.3000 | 10.6333 | 12.3333 | 9.3667 |
| 5 | 17.9667 | 19.6667 | 15.2000 | 14.3333 | 14.0000 | 13.4667 | 12.0000 | 10.3667 |
| 4 | 23.0667 | 21.2000 | 19.2667 | 19.4333 | 15.8333 | 17.0667 | 16.2000 | 12.2333 |
| 3 | 29.2333 | 29.5000 | 25.0667 | 22.4333 | 20.0333 | 22.5333 | 19.1333 | 18.1667 |
| 2 | 41.3000 | 37.8000 | 42.3000 | 34.3000 | 32.2333 | 31.6667 | 26.6000 | 23.3333 |
| 1 | 73.0667 | 84.5333 | 72.5667 | 65.2000 | 52.9333 | 59.0000 | 62.0000 | 48.0667 |

Table A.14: Reduce-Max and symptoms appearance for Uniform Part 1

| FVMs | 16 h | 15 h | 14 h | 13 h | 12 h | 11 h | 10 h | 9 h |
|---|---|---|---|---|---|---|---|---|
| 1 | 0.1573 | 0.1521 | 0.1500 | 0.1516 | 0.1487 | 0.1499 | 0.1487 | 0.1450 |
| 2 | 0.0801 | 0.0809 | 0.0784 | 0.0775 | 0.0785 | 0.0775 | 0.0767 | 0.0766 |
| 3 | 0.0565 | 0.0548 | 0.0546 | 0.0544 | 0.0540 | 0.0548 | 0.0541 | 0.0542 |
| 4 | 0.0426 | 0.0418 | 0.0424 | 0.0421 | 0.0423 | 0.0423 | 0.0428 | 0.0428 |
| 5 | 0.0350 | 0.0353 | 0.0352 | 0.0352 | 0.0355 | 0.0355 | 0.0357 | 0.0361 |
| 6 | 0.0313 | 0.0310 | 0.0304 | 0.0301 | 0.0310 | 0.0307 | 0.0312 | 0.0316 |
| 7 | 0.0272 | 0.0264 | 0.0268 | 0.0268 | 0.0269 | 0.0273 | 0.0279 | 0.0288 |
| 8 | 0.0240 | 0.0239 | 0.0243 | 0.0251 | 0.0250 | 0.0251 | 0.0259 | 0.0263 |
| 9 | 0.0219 | 0.0217 | 0.0223 | 0.0224 | 0.0230 | 0.0231 | 0.0236 | 0.0245 |
| 10 | 0.0204 | 0.0204 | 0.0206 | 0.0208 | 0.0211 | 0.0215 | 0.0223 | 0.0231 |
| 11 | 0.0188 | 0.0191 | 0.0193 | 0.0196 | 0.0204 | 0.0203 | 0.0210 | 0.0219 |
| 12 | 0.0177 | 0.0184 | 0.0181 | 0.0185 | 0.0189 | 0.0194 | 0.0202 | 0.0208 |
| 13 | 0.0168 | 0.0169 | 0.0172 | 0.0174 | 0.0180 | 0.0188 | 0.0192 | 0.0203 |
| 14 | 0.0160 | 0.0163 | 0.0165 | 0.0168 | 0.0172 | 0.0180 | 0.0187 | 0.0197 |
| 15 | 0.0152 | 0.0158 | 0.0158 | 0.0163 | 0.0168 | 0.0171 | 0.0181 | 0.0189 |
| 16 | 0.0147 | 0.0150 | 0.0151 | 0.0156 | 0.0160 | 0.0168 | 0.0176 | 0.0185 |

Table A.15: Reduce-Max and symptoms appearance for Uniform Part 2

| FVMs | 8 h | 7 h | 6 h | 5 h | 4 h | 3 h | 2 h |
|---|---|---|---|---|---|---|---|
| 1 | 0.1432 | 0.1422 | 0.1410 | 0.1391 | 0.1380 | 0.1362 | 0.1354 |
| 2 | 0.0765 | 0.0763 | 0.0775 | 0.0781 | 0.0802 | 0.0837 | 0.0927 |
| 3 | 0.0545 | 0.0549 | 0.0560 | 0.0579 | 0.0606 | 0.0656 | 0.0788 |
| 4 | 0.0435 | 0.0444 | 0.0456 | 0.0480 | 0.0510 | 0.0581 | 0.0732 |
| 5 | 0.0367 | 0.0380 | 0.0395 | 0.0419 | 0.0460 | 0.0534 | 0.0702 |
| 6 | 0.0326 | 0.0338 | 0.0354 | 0.0382 | 0.0423 | 0.0503 | 0.0688 |
| 7 | 0.0296 | 0.0306 | 0.0326 | 0.0354 | 0.0401 | 0.0486 | 0.0678 |
| 8 | 0.0272 | 0.0287 | 0.0305 | 0.0335 | 0.0383 | 0.0474 | 0.0676 |
| 9 | 0.0255 | 0.0271 | 0.0292 | 0.0322 | 0.0372 | 0.0465 | 0.0673 |
| 10 | 0.0242 | 0.0256 | 0.0281 | 0.0311 | 0.0365 | 0.0461 | 0.0671 |
| 11 | 0.0231 | 0.0249 | 0.0270 | 0.0303 | 0.0359 | 0.0456 | 0.0671 |
| 12 | 0.0223 | 0.0238 | 0.0261 | 0.0298 | 0.0354 | 0.0454 | 0.0669 |
| 13 | 0.0216 | 0.0232 | 0.0257 | 0.0292 | 0.0350 | 0.0452 | 0.0668 |
| 14 | 0.0208 | 0.0227 | 0.0249 | 0.0287 | 0.0346 | 0.0451 | 0.0668 |
| 15 | 0.0203 | 0.0223 | 0.0247 | 0.0285 | 0.0344 | 0.0449 | 0.0668 |
| 16 | 0.0198 | 0.0217 | 0.0242 | 0.0284 | 0.0342 | 0.0449 | 0.0668 |

Table A.16: Reduce-Max and symptoms appearance for Random Part 1

| FVMs | 16 h | 15 h | 14 h | 13 h | 12 h | 11 h | 10 h | 9 h |
|---|---|---|---|---|---|---|---|---|
| 1 | 0.1783 | 0.1714 | 0.1791 | 0.1715 | 0.1708 | 0.1644 | 0.1665 | 0.1655 |
| 2 | 0.0900 | 0.0966 | 0.0906 | 0.0892 | 0.0922 | 0.0875 | 0.0862 | 0.0878 |
| 3 | 0.0608 | 0.0635 | 0.0610 | 0.0605 | 0.0604 | 0.0612 | 0.0608 | 0.0607 |
| 4 | 0.0468 | 0.0489 | 0.0471 | 0.0468 | 0.0474 | 0.0467 | 0.0479 | 0.0491 |
| 5 | 0.0416 | 0.0389 | 0.0396 | 0.0399 | 0.0400 | 0.0392 | 0.0403 | 0.0416 |
| 6 | 0.0345 | 0.0339 | 0.0344 | 0.0339 | 0.0338 | 0.0344 | 0.0366 | 0.0365 |
| 7 | 0.0294 | 0.0297 | 0.0303 | 0.0316 | 0.0296 | 0.0312 | 0.0312 | 0.0324 |
| 8 | 0.0272 | 0.0271 | 0.0267 | 0.0271 | 0.0282 | 0.0275 | 0.0283 | 0.0293 |
| 9 | 0.0244 | 0.0252 | 0.0248 | 0.0251 | 0.0253 | 0.0261 | 0.0267 | 0.0275 |
| 10 | 0.0225 | 0.0229 | 0.0228 | 0.0231 | 0.0232 | 0.0238 | 0.0248 | 0.0259 |
| 11 | 0.0213 | 0.0208 | 0.0210 | 0.0216 | 0.0224 | 0.0228 | 0.0236 | 0.0246 |
| 12 | 0.0195 | 0.0197 | 0.0202 | 0.0203 | 0.0212 | 0.0216 | 0.0224 | 0.0237 |
| 13 | 0.0189 | 0.0195 | 0.0188 | 0.0195 | 0.0203 | 0.0207 | 0.0217 | 0.0229 |
| 14 | 0.0179 | 0.0181 | 0.0186 | 0.0190 | 0.0197 | 0.0199 | 0.0209 | 0.0219 |
| 15 | 0.0169 | 0.0171 | 0.0184 | 0.0181 | 0.0189 | 0.0194 | 0.0204 | 0.0217 |
| 16 | 0.0162 | 0.0166 | 0.0168 | 0.0175 | 0.0179 | 0.0187 | 0.0196 | 0.0210 |

Table A.17: Reduce-Max and symptoms appearance for Random Part 2

| FVMs | 8 h | 7 h | 6 h | 5 h | 4 h | 3 h | 2 h |
|---|---|---|---|---|---|---|---|
| 1 | 0.1671 | 0.1662 | 0.1590 | 0.1590 | 0.1596 | 0.1596 | 0.1569 |
| 2 | 0.0856 | 0.0870 | 0.0888 | 0.0909 | 0.0907 | 0.0934 | 0.1031 |
| 3 | 0.0599 | 0.0622 | 0.0641 | 0.0662 | 0.0681 | 0.0758 | 0.0888 |
| 4 | 0.0494 | 0.0498 | 0.0515 | 0.0542 | 0.0577 | 0.0652 | 0.0834 |
| 5 | 0.0416 | 0.0429 | 0.0450 | 0.0475 | 0.0518 | 0.0598 | 0.0804 |
| 6 | 0.0357 | 0.0374 | 0.0396 | 0.0425 | 0.0485 | 0.0574 | 0.0794 |
| 7 | 0.0334 | 0.0351 | 0.0371 | 0.0397 | 0.0454 | 0.0555 | 0.0784 |
| 8 | 0.0303 | 0.0323 | 0.0340 | 0.0382 | 0.0430 | 0.0546 | 0.0783 |
| 9 | 0.0297 | 0.0302 | 0.0324 | 0.0363 | 0.0425 | 0.0533 | 0.0786 |
| 10 | 0.0269 | 0.0292 | 0.0316 | 0.0355 | 0.0415 | 0.0529 | 0.0774 |
| 11 | 0.0260 | 0.0277 | 0.0302 | 0.0347 | 0.0412 | 0.0528 | 0.0786 |
| 12 | 0.0254 | 0.0268 | 0.0296 | 0.0337 | 0.0404 | 0.0532 | 0.0774 |
| 13 | 0.0241 | 0.0258 | 0.0289 | 0.0336 | 0.0401 | 0.0524 | 0.0778 |
| 14 | 0.0235 | 0.0256 | 0.0288 | 0.0329 | 0.0395 | 0.0523 | 0.0781 |
| 15 | 0.0231 | 0.0250 | 0.0278 | 0.0322 | 0.0395 | 0.0532 | 0.0781 |
| 16 | 0.0223 | 0.0249 | 0.0278 | 0.0325 | 0.0396 | 0.0527 | 0.0783 |

Table A.18: Reduce-Max and symptoms appearance for Arithmetic Part 1

| FVMs | 16 h | 15 h | 14 h | 13 h | 12 h | 11 h | 10 h | 9 h |
|---|---|---|---|---|---|---|---|---|
| 1 | 0.1698 | 0.1678 | 0.1652 | 0.1622 | 0.1719 | 0.1642 | 0.1562 | 0.1615 |
| 2 | 0.0872 | 0.0858 | 0.0911 | 0.0857 | 0.0849 | 0.0835 | 0.0827 | 0.0810 |
| 3 | 0.0592 | 0.0593 | 0.0588 | 0.0595 | 0.0592 | 0.0593 | 0.0574 | 0.0601 |
| 4 | 0.0481 | 0.0459 | 0.0460 | 0.0456 | 0.0456 | 0.0448 | 0.0454 | 0.0457 |
| 5 | 0.0384 | 0.0385 | 0.0378 | 0.0381 | 0.0381 | 0.0382 | 0.0379 | 0.0377 |
| 6 | 0.0322 | 0.0326 | 0.0322 | 0.0327 | 0.0328 | 0.0332 | 0.0328 | 0.0334 |
| 7 | 0.0286 | 0.0291 | 0.0294 | 0.0290 | 0.0292 | 0.0294 | 0.0291 | 0.0295 |
| 8 | 0.0261 | 0.0271 | 0.0256 | 0.0264 | 0.0263 | 0.0262 | 0.0273 | 0.0273 |
| 9 | 0.0235 | 0.0239 | 0.0236 | 0.0239 | 0.0238 | 0.0245 | 0.0248 | 0.0256 |
| 10 | 0.0224 | 0.0224 | 0.0220 | 0.0226 | 0.0223 | 0.0225 | 0.0231 | 0.0241 |
| 11 | 0.0206 | 0.0201 | 0.0213 | 0.0209 | 0.0211 | 0.0214 | 0.0221 | 0.0225 |
| 12 | 0.0191 | 0.0192 | 0.0195 | 0.0195 | 0.0209 | 0.0203 | 0.0210 | 0.0222 |
| 13 | 0.0180 | 0.0187 | 0.0185 | 0.0194 | 0.0201 | 0.0196 | 0.0203 | 0.0208 |
| 14 | 0.0177 | 0.0174 | 0.0177 | 0.0185 | 0.0182 | 0.0188 | 0.0193 | 0.0204 |
| 15 | 0.0166 | 0.0162 | 0.0170 | 0.0182 | 0.0178 | 0.0180 | 0.0186 | 0.0198 |
| 16 | 0.0157 | 0.0158 | 0.0164 | 0.0166 | 0.0169 | 0.0176 | 0.0182 | 0.0192 |

Table A.19: Reduce-Max and symptoms appearance for Arithmetic Part 2

| FVMs | 8 h | 7 h | 6 h | 5 h | 4 h | 3 h | 2 h |
|---|---|---|---|---|---|---|---|
| 1 | 0.1511 | 0.1471 | 0.1445 | 0.1440 | 0.1421 | 0.1393 | 0.1370 |
| 2 | 0.0796 | 0.0808 | 0.0798 | 0.0804 | 0.0811 | 0.0845 | 0.0932 |
| 3 | 0.0572 | 0.0577 | 0.0575 | 0.0587 | 0.0616 | 0.0668 | 0.0790 |
| 4 | 0.0449 | 0.0457 | 0.0467 | 0.0487 | 0.0518 | 0.0580 | 0.0734 |
| 5 | 0.0390 | 0.0392 | 0.0403 | 0.0426 | 0.0465 | 0.0535 | 0.0707 |
| 6 | 0.0335 | 0.0350 | 0.0367 | 0.0386 | 0.0427 | 0.0507 | 0.0691 |
| 7 | 0.0304 | 0.0314 | 0.0332 | 0.0359 | 0.0409 | 0.0489 | 0.0687 |
| 8 | 0.0287 | 0.0294 | 0.0310 | 0.0340 | 0.0392 | 0.0483 | 0.0684 |
| 9 | 0.0267 | 0.0280 | 0.0300 | 0.0332 | 0.0382 | 0.0473 | 0.0681 |
| 10 | 0.0252 | 0.0265 | 0.0284 | 0.0315 | 0.0372 | 0.0469 | 0.0682 |
| 11 | 0.0235 | 0.0251 | 0.0274 | 0.0308 | 0.0367 | 0.0465 | 0.0682 |
| 12 | 0.0227 | 0.0244 | 0.0271 | 0.0300 | 0.0363 | 0.0463 | 0.0682 |
| 13 | 0.0221 | 0.0238 | 0.0269 | 0.0300 | 0.0358 | 0.0462 | 0.0681 |
| 14 | 0.0214 | 0.0233 | 0.0262 | 0.0294 | 0.0357 | 0.0461 | 0.0681 |
| 15 | 0.0213 | 0.0230 | 0.0263 | 0.0294 | 0.0359 | 0.0462 | 0.0681 |
| 16 | 0.0203 | 0.0228 | 0.0262 | 0.0293 | 0.0355 | 0.0460 | 0.0683 |

Table A.20: Reduce-Max and symptoms appearance for Harmonic Part 1

| FVMs | 16 h | 15 h | 14 h | 13 h | 12 h | 11 h | 10 h | 9 h |
|---|---|---|---|---|---|---|---|---|
| 1 | 3.2405 | 1.7934 | 2.5924 | 1.8644 | 1.9709 | 2.0952 | 1.7046 | 1.4294 |
| 2 | 1.2074 | 1.1009 | 1.1009 | 1.0476 | 1.0565 | 1.1719 | 0.9233 | 0.7014 |
| 3 | 0.8434 | 0.9588 | 0.7901 | 0.8168 | 0.7014 | 0.5771 | 0.5238 | 0.5504 |
| 4 | 0.7458 | 0.7102 | 0.5149 | 0.5593 | 0.5149 | 0.4705 | 0.4794 | 0.4972 |
| 5 | 0.5682 | 0.5504 | 0.4972 | 0.4173 | 0.4350 | 0.4084 | 0.3285 | 0.2930 |
| 6 | 0.4705 | 0.4439 | 0.4528 | 0.3818 | 0.5149 | 0.3019 | 0.3729 | 0.3906 |
| 7 | 0.4972 | 0.4261 | 0.3196 | 0.3551 | 0.3906 | 0.2752 | 0.2841 | 0.2486 |
| 8 | 0.3729 | 0.3551 | 0.3818 | 0.2841 | 0.2575 | 0.2841 | 0.2397 | 0.2308 |
| 9 | 0.3107 | 0.3551 | 0.3019 | 0.4439 | 0.2308 | 0.3019 | 0.2131 | 0.1864 |
| 10 | 0.3374 | 0.3196 | 0.2663 | 0.2308 | 0.2308 | 0.2308 | 0.2131 | 0.2131 |
| 11 | 0.2841 | 0.2486 | 0.2841 | 0.2131 | 0.2220 | 0.1953 | 0.1776 | 0.1776 |
| 12 | 0.2841 | 0.2220 | 0.2486 | 0.1953 | 0.1953 | 0.1598 | 0.1598 | 0.1598 |
| 13 | 0.2486 | 0.2486 | 0.2308 | 0.2308 | 0.1953 | 0.1687 | 0.1598 | 0.1598 |
| 14 | 0.2220 | 0.2486 | 0.2131 | 0.1953 | 0.1598 | 0.1687 | 0.1509 | 0.1332 |
| 15 | 0.2308 | 0.1953 | 0.1776 | 0.1687 | 0.1598 | 0.1598 | 0.1332 | 0.1243 |
| 16 | 0.2397 | 0.1687 | 0.1687 | 0.1598 | 0.1420 | 0.1243 | 0.1243 | 0.1065 |

Table A.21: Reduce-Max and symptoms appearance for Harmonic Part 2

| FVMs | 8 h | 7 h | 6 h | 5 h | 4 h | 3 h | 2 h |
|---|---|---|---|---|---|---|---|
| 1 | 1.1985 | 1.0387 | 1.0032 | 0.8878 | 0.7014 | 0.5416 | 0.4439 |
| 2 | 0.7014 | 0.6392 | 0.5327 | 0.5682 | 0.3551 | 0.3019 | 0.1953 |
| 3 | 0.4617 | 0.4350 | 0.4084 | 0.2841 | 0.2841 | 0.2486 | 0.1598 |
| 4 | 0.3374 | 0.3107 | 0.3196 | 0.2663 | 0.2131 | 0.1687 | 0.1243 |
| 5 | 0.3285 | 0.2575 | 0.2220 | 0.1776 | 0.1687 | 0.1243 | 0.1243 |
| 6 | 0.2841 | 0.2308 | 0.2663 | 0.1509 | 0.1332 | 0.1154 | 0.1065 |
| 7 | 0.2308 | 0.2663 | 0.1776 | 0.1776 | 0.1332 | 0.1065 | 0.1065 |
| 8 | 0.2131 | 0.1776 | 0.1509 | 0.1509 | 0.1065 | 0.0888 | 0.0888 |
| 9 | 0.1776 | 0.1687 | 0.1598 | 0.1243 | 0.1065 | 0.1065 | 0.0888 |
| 10 | 0.1776 | 0.1420 | 0.1420 | 0.1154 | 0.0888 | 0.0888 | 0.0888 |
| 11 | 0.1420 | 0.1332 | 0.1598 | 0.0888 | 0.0888 | 0.0888 | 0.0888 |
| 12 | 0.1953 | 0.1243 | 0.1154 | 0.0888 | 0.0888 | 0.0710 | 0.0888 |
| 13 | 0.1598 | 0.1332 | 0.1065 | 0.0799 | 0.0888 | 0.0888 | 0.0888 |
| 14 | 0.1243 | 0.1243 | 0.0977 | 0.0888 | 0.0888 | 0.0710 | 0.0888 |
| 15 | 0.1154 | 0.1065 | 0.0888 | 0.0888 | 0.0710 | 0.0710 | 0.0888 |
| 16 | 0.1065 | 0.1065 | 0.0888 | 0.0888 | 0.0710 | 0.0710 | 0.0888 |

Table A.22: Reduce-Max and symptoms appearance for Exponential Part 1

| FVMs | 16 h | 15 h | 14 h | 13 h | 12 h | 11 h | 10 h | 9 h |
|---|---|---|---|---|---|---|---|---|
| 1 | 8.8333 | 7.9333 | 7.4667 | 7.2000 | 6.0000 | 6.5667 | 6.5333 | 5.0000 |
| 2 | 4.5667 | 4.3333 | 4.8667 | 3.9333 | 3.5000 | 3.9333 | 2.9667 | 3.0000 |
| 3 | 3.1333 | 3.1333 | 2.9333 | 2.6000 | 2.5333 | 2.1667 | 2.4000 | 2.1667 |
| 4 | 2.6667 | 2.3667 | 2.7333 | 2.4000 | 1.8000 | 1.7667 | 1.6667 | 1.4000 |
| 5 | 2.0667 | 2.2667 | 1.6000 | 1.6667 | 1.7333 | 1.5333 | 1.2667 | 1.3333 |
| 6 | 1.9667 | 1.8333 | 1.8333 | 1.3333 | 1.4667 | 1.4667 | 1.0333 | 1.1333 |
| 7 | 1.4000 | 1.3667 | 1.5667 | 1.2500 | 1.4000 | 1.1000 | 1.0000 | 0.8667 |
| 8 | 1.4667 | 1.3333 | 1.4000 | 1.2000 | 1.1000 | 1.0000 | 0.7667 | 0.7333 |
| 9 | 1.2667 | 1.0667 | 1.0000 | 1.0667 | 0.8667 | 0.8667 | 0.8667 | 0.7000 |
| 10 | 1.2333 | 1.3333 | 1.0667 | 1.0000 | 0.8667 | 0.8000 | 0.8000 | 0.7000 |
| 11 | 1.0000 | 1.0667 | 0.8667 | 0.8667 | 0.8000 | 0.8000 | 0.7333 | 0.5667 |
| 12 | 1.0000 | 0.9000 | 1.0000 | 0.7333 | 0.6333 | 0.8000 | 0.6333 | 0.6667 |
| 13 | 0.9333 | 0.9333 | 0.7333 | 0.8667 | 0.7333 | 0.5667 | 0.5667 | 0.6333 |
| 14 | 1.1333 | 0.7333 | 0.8000 | 0.6333 | 0.7333 | 0.5333 | 0.5667 | 0.4667 |
| 15 | 0.8000 | 0.8000 | 0.7000 | 0.6667 | 0.6000 | 0.5333 | 0.5000 | 0.4333 |
| 16 | 0.6667 | 0.7000 | 0.7333 | 0.5333 | 0.4667 | 0.4667 | 0.4667 | 0.4667 |

Table A.23: Reduce-Max and symptoms appearance for Exponential Part 2

| FVMs | 8 h | 7 h | 6 h | 5 h | 4 h | 3 h | 2 h |
|---|---|---|---|---|---|---|---|
| 1 | 5.4000 | 3.8000 | 3.7333 | 2.8667 | 2.3333 | 1.5333 | 1.0000 |
| 2 | 2.4000 | 2.7333 | 1.8000 | 1.3667 | 1.2000 | 0.8333 | 0.5333 |
| 3 | 1.9667 | 1.7667 | 1.2333 | 1.0667 | 0.9333 | 0.7333 | 0.4000 |
| 4 | 1.4667 | 1.3333 | 1.0000 | 0.8000 | 0.7333 | 0.5333 | 0.3000 |
| 5 | 1.0000 | 0.9333 | 0.7667 | 0.7333 | 0.6000 | 0.4000 | 0.2667 |
| 6 | 0.9333 | 0.7333 | 0.7333 | 0.5667 | 0.5333 | 0.3333 | 0.2667 |
| 7 | 0.8000 | 0.7000 | 0.6667 | 0.5000 | 0.4000 | 0.3000 | 0.2000 |
| 8 | 0.7667 | 0.6333 | 0.6000 | 0.4667 | 0.4000 | 0.3333 | 0.2000 |
| 9 | 0.6000 | 0.5333 | 0.4667 | 0.4000 | 0.3333 | 0.2333 | 0.1667 |
| 10 | 0.7333 | 0.5333 | 0.5000 | 0.3333 | 0.3000 | 0.2667 | 0.1667 |
| 11 | 0.5667 | 0.5667 | 0.4000 | 0.3000 | 0.3000 | 0.2000 | 0.1333 |
| 12 | 0.5667 | 0.4667 | 0.4000 | 0.3333 | 0.2667 | 0.2333 | 0.1333 |
| 13 | 0.5333 | 0.4333 | 0.3667 | 0.3000 | 0.2667 | 0.2000 | 0.1333 |
| 14 | 0.4000 | 0.4000 | 0.3333 | 0.2667 | 0.2667 | 0.2000 | 0.1333 |
| 15 | 0.4667 | 0.3333 | 0.2667 | 0.2667 | 0.2000 | 0.2000 | 0.1333 |
| 16 | 0.4000 | 0.3333 | 0.2667 | 0.2667 | 0.2000 | 0.1333 | 0.1333 |

Table A.24: Total-max of Uniform weights for different numbers of clusters

| | 1 Cluster | 2 Clusters | 4 Clusters | 8 Clusters | 16 Clusters |
|---|---|---|---|---|---|
| determ. | 0.062439 | 0.062439 | 0.062439 | 0.062439 | 0.062439 |
| $\alpha$=1 | 0.678556 | 0.351941 | 0.188976 | 0.133268 | 0.138498 |
| $\alpha$=1.2 | 0.701415 | 0.368156 | 0.197561 | 0.133863 | 0.138156 |
| $\alpha$=1.4 | 0.721873 | 0.375073 | 0.204839 | 0.135951 | 0.138 |
| $\alpha$=1.6 | 0.740039 | 0.387278 | 0.212507 | 0.135941 | 0.137922 |
| $\alpha$=1.8 | 0.757112 | 0.393805 | 0.212146 | 0.135941 | 0.138273 |
| $\alpha$=2 | 0.777054 | 0.404566 | 0.218488 | 0.141102 | 0.132488 |
| $\alpha$=2.2 | 0.789629 | 0.414049 | 0.225327 | 0.141044 | 0.132956 |
| $\alpha$=2.4 | 0.801844 | 0.418507 | 0.230868 | 0.147239 | 0.132673 |
| $\alpha$=2.6 | 0.812849 | 0.427649 | 0.235737 | 0.146839 | 0.132478 |
| $\alpha$=2.8 | 0.827893 | 0.434488 | 0.240566 | 0.153073 | 0.1376 |
| $\alpha$=3 | 0.836927 | 0.439112 | 0.241346 | 0.152956 | 0.137493 |

Table A.25: Total-max for Poisson (2) for different numbers of clusters

| | 1 Cluster | 2 Clusters | 4 Clusters | 8 Clusters | 16 Clusters |
|---|---|---|---|---|---|
| determ. | 0.078026 | 0.079164 | 0.081564 | 0.084107 | 0.088422 |
| $\alpha$=1 | 0.799743 | 0.414708 | 0.225685 | 0.174646 | 0.192779 |
| $\alpha$=1.2 | 0.826375 | 0.434799 | 0.237255 | 0.175315 | 0.192329 |
| $\alpha$=1.4 | 0.852324 | 0.441241 | 0.250305 | 0.189951 | 0.193236 |
| $\alpha$=1.6 | 0.872678 | 0.456965 | 0.259481 | 0.189101 | 0.192411 |
| $\alpha$=1.8 | 0.890198 | 0.463414 | 0.259412 | 0.188392 | 0.193091 |
| $\alpha$=2 | 0.912695 | 0.476663 | 0.271609 | 0.205379 | 0.210437 |
| $\alpha$=2.2 | 0.931868 | 0.488025 | 0.281871 | 0.208598 | 0.214584 |
| $\alpha$=2.4 | 0.946664 | 0.495657 | 0.291145 | 0.222825 | 0.21157 |
| $\alpha$=2.6 | 0.959218 | 0.506113 | 0.300741 | 0.226575 | 0.211962 |
| $\alpha$=2.8 | 0.975769 | 0.513426 | 0.30827 | 0.241572 | 0.239899 |
| $\alpha$=3 | 0.98526 | 0.519815 | 0.310706 | 0.242131 | 0.241959 |

Table A.26: Total-max for Poisson (8) for different numbers of clusters

| | 1 Cluster | 2 Clusters | 4 Clusters | 8 Clusters | 16 Clusters |
|---|---|---|---|---|---|
| determ. | 0.073852 | 0.074897 | 0.076048 | 0.077227 | 0.078418 |
| $\alpha$=1 | 0.757381 | 0.390918 | 0.205601 | 0.148066 | 0.155819 |
| $\alpha$=1.2 | 0.783175 | 0.407603 | 0.215252 | 0.14862 | 0.154896 |
| $\alpha$=1.4 | 0.806835 | 0.414648 | 0.223208 | 0.153651 | 0.15593 |
| $\alpha$=1.6 | 0.826209 | 0.429904 | 0.230531 | 0.153902 | 0.155714 |
| $\alpha$=1.8 | 0.844807 | 0.435013 | 0.23056 | 0.153549 | 0.155734 |
| $\alpha$=2 | 0.867796 | 0.446775 | 0.237542 | 0.164105 | 0.163973 |
| $\alpha$=2.2 | 0.881444 | 0.455628 | 0.246068 | 0.163128 | 0.160977 |
| $\alpha$=2.4 | 0.894289 | 0.461232 | 0.251749 | 0.172773 | 0.161925 |
| $\alpha$=2.6 | 0.906939 | 0.469396 | 0.259062 | 0.171789 | 0.158693 |
| $\alpha$=2.8 | 0.922057 | 0.476249 | 0.264842 | 0.183248 | 0.17435 |
| $\alpha$=3 | 0.932965 | 0.483064 | 0.264173 | 0.183426 | 0.173306 |

Table A.27: Total-max for Random (2) for different numbers of clusters

| | 1 Cluster | 2 Clusters | 4 Clusters | 8 Clusters | 16 Clusters |
|---|---|---|---|---|---|
| determ. | 0.078967 | 0.080057 | 0.079687 | 0.080069 | 0.081219 |
| $\alpha$=1 | 0.816897 | 0.419744 | 0.218132 | 0.146863 | 0.151502 |
| $\alpha$=1.2 | 0.844087 | 0.436563 | 0.225402 | 0.147138 | 0.152663 |
| $\alpha$=1.4 | 0.868734 | 0.44326 | 0.233771 | 0.150376 | 0.151989 |
| $\alpha$=1.6 | 0.890405 | 0.458628 | 0.240981 | 0.151071 | 0.151963 |
| $\alpha$=1.8 | 0.906506 | 0.465589 | 0.241667 | 0.150675 | 0.151614 |
| $\alpha$=2 | 0.933613 | 0.476948 | 0.247869 | 0.157704 | 0.153606 |
| $\alpha$=2.2 | 0.948883 | 0.487896 | 0.25198 | 0.157931 | 0.153867 |
| $\alpha$=2.4 | 0.963471 | 0.493544 | 0.258333 | 0.166035 | 0.153897 |
| $\alpha$=2.6 | 0.97775 | 0.502446 | 0.262364 | 0.164939 | 0.154471 |
| $\alpha$=2.8 | 0.992884 | 0.512556 | 0.26903 | 0.174976 | 0.164637 |
| $\alpha$=3 | 1.002645 | 0.515227 | 0.267988 | 0.176874 | 0.165084 |

Table A.28: Total-max for Random (8) for different numbers of clusters

| | 1 Cluster | 2 Clusters | 4 Clusters | 8 Clusters | 16 Clusters |
|---|---|---|---|---|---|
| determ. | 0.074657 | 0.075363 | 0.077586 | 0.079219 | 0.083438 |
| $\alpha$=1 | 0.755542 | 0.393123 | 0.208919 | 0.151026 | 0.160605 |
| $\alpha$=1.2 | 0.780464 | 0.40981 | 0.218265 | 0.151024 | 0.16104 |
| $\alpha$=1.4 | 0.80314 | 0.417302 | 0.227437 | 0.157294 | 0.161122 |
| $\alpha$=1.6 | 0.82379 | 0.431001 | 0.235308 | 0.157254 | 0.161176 |
| $\alpha$=1.8 | 0.840548 | 0.437356 | 0.234659 | 0.157788 | 0.161439 |
| $\alpha$=2 | 0.862938 | 0.449886 | 0.241388 | 0.165739 | 0.165757 |
| $\alpha$=2.2 | 0.877122 | 0.461077 | 0.249491 | 0.166559 | 0.165078 |
| $\alpha$=2.4 | 0.890487 | 0.464793 | 0.256607 | 0.178962 | 0.164789 |
| $\alpha$=2.6 | 0.907131 | 0.475585 | 0.263437 | 0.176228 | 0.164787 |
| $\alpha$=2.8 | 0.919962 | 0.482504 | 0.270859 | 0.187634 | 0.179648 |
| $\alpha$=3 | 0.931583 | 0.487383 | 0.270514 | 0.186337 | 0.178445 |

Table A.29: Balcancing Mobility Algorithm for Uniform Part 1

| FVM | 2 h | 3 h | 4 h | 5 h | 6 h | 7 h | 8 h | 9 h |
|---|---|---|---|---|---|---|---|---|
| 16 | 0.07 | 0.04 | 0.03 | 0.03 | 0.02 | 0.02 | 0.02 | 0.02 |
| 15 | 0.07 | 0.04 | 0.03 | 0.03 | 0.02 | 0.02 | 0.02 | 0.02 |
| 14 | 0.07 | 0.05 | 0.03 | 0.03 | 0.03 | 0.02 | 0.02 | 0.02 |
| 13 | 0.07 | 0.05 | 0.03 | 0.03 | 0.03 | 0.02 | 0.02 | 0.02 |
| 12 | 0.07 | 0.05 | 0.04 | 0.03 | 0.03 | 0.02 | 0.02 | 0.02 |
| 11 | 0.07 | 0.05 | 0.04 | 0.03 | 0.03 | 0.02 | 0.02 | 0.02 |
| 10 | 0.07 | 0.05 | 0.04 | 0.03 | 0.03 | 0.03 | 0.02 | 0.02 |
| 9 | 0.07 | 0.05 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 | 0.02 |
| 8 | 0.07 | 0.05 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 |
| 7 | 0.07 | 0.05 | 0.04 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 |
| 6 | 0.07 | 0.05 | 0.04 | 0.04 | 0.04 | 0.03 | 0.03 | 0.03 |
| 5 | 0.07 | 0.05 | 0.05 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 |
| 4 | 0.07 | 0.06 | 0.05 | 0.05 | 0.05 | 0.04 | 0.04 | 0.04 |
| 3 | 0.08 | 0.07 | 0.06 | 0.06 | 0.06 | 0.06 | 0.05 | 0.05 |
| 2 | 0.09 | 0.08 | 0.08 | 0.08 | 0.08 | 0.08 | 0.08 | 0.08 |
| 1 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 |

Table A.30: Balcancing Mobility Algorithm for Uniform Part 2

| FVM | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|
| 16 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.01 | 0.01 |
| 15 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 14 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 13 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 12 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 11 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 10 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 9 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 8 | 0.03 | 0.03 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 7 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 |
| 6 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 |
| 5 | 0.04 | 0.04 | 0.04 | 0.04 | 0.03 | 0.03 | 0.03 |
| 4 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 |
| 3 | 0.05 | 0.05 | 0.05 | 0.05 | 0.05 | 0.06 | 0.06 |
| 2 | 0.08 | 0.08 | 0.08 | 0.08 | 0.08 | 0.08 | 0.08 |
| 1 | 0.15 | 0.15 | 0.15 | 0.15 | 0.15 | 0.16 | 0.16 |

Table A.31: Balcancing Mobility Algorithm for Random Part 1

| | 2 h | 3 h | 4 h | 5 h | 6 h | 7 h | 8 h | 9 h |
|---|---|---|---|---|---|---|---|---|
| 16 | 0.08 | 0.05 | 0.04 | 0.03 | 0.03 | 0.02 | 0.02 | 0.02 |
| 15 | 0.08 | 0.05 | 0.04 | 0.03 | 0.03 | 0.03 | 0.02 | 0.02 |
| 14 | 0.08 | 0.05 | 0.04 | 0.03 | 0.03 | 0.03 | 0.02 | 0.02 |
| 13 | 0.08 | 0.05 | 0.04 | 0.03 | 0.03 | 0.03 | 0.02 | 0.02 |
| 12 | 0.08 | 0.05 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 | 0.02 |
| 11 | 0.08 | 0.05 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 | 0.02 |
| 10 | 0.08 | 0.05 | 0.04 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 |
| 9 | 0.08 | 0.05 | 0.04 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 |
| 8 | 0.08 | 0.05 | 0.04 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 |
| 7 | 0.08 | 0.06 | 0.05 | 0.04 | 0.04 | 0.03 | 0.03 | 0.03 |
| 6 | 0.08 | 0.06 | 0.05 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 |
| 5 | 0.08 | 0.06 | 0.05 | 0.05 | 0.04 | 0.04 | 0.04 | 0.04 |
| 4 | 0.08 | 0.06 | 0.06 | 0.05 | 0.05 | 0.05 | 0.05 | 0.05 |
| 3 | 0.09 | 0.07 | 0.07 | 0.06 | 0.06 | 0.06 | 0.06 | 0.06 |
| 2 | 0.11 | 0.10 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 |
| 1 | 0.16 | 0.16 | 0.16 | 0.16 | 0.16 | 0.16 | 0.16 | 0.17 |

Table A.32: Balcancing Mobility Algorithm for Random Part 2

| | 10 h | 11 h | 12 h | 13 h | 14 h | 15 h | 16 h |
|---|---|---|---|---|---|---|---|
| 16 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 15 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 14 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 13 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 12 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 11 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 10 | 0.03 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 9 | 0.03 | 0.03 | 0.03 | 0.02 | 0.02 | 0.03 | 0.02 |
| 8 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 |
| 7 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 |
| 6 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 |
| 5 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 |
| 4 | 0.05 | 0.05 | 0.05 | 0.05 | 0.05 | 0.05 | 0.05 |
| 3 | 0.06 | 0.06 | 0.06 | 0.06 | 0.06 | 0.06 | 0.07 |
| 2 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 |
| 1 | 0.16 | 0.16 | 0.17 | 0.17 | 0.17 | 0.17 | 0.17 |

Table A.33: Balcancing Mobility Algorithm for Poisson Part 1

| | 2 h | 3 h | 4 h | 5 h | 6 h | 7 h | 8 h | 9 h |
|---|---|---|---|---|---|---|---|---|
| 16 | 0.07 | 0.04 | 0.03 | 0.03 | 0.02 | 0.02 | 0.02 | 0.02 |
| 15 | 0.07 | 0.04 | 0.03 | 0.03 | 0.02 | 0.02 | 0.02 | 0.02 |
| 14 | 0.07 | 0.05 | 0.03 | 0.03 | 0.03 | 0.02 | 0.02 | 0.02 |
| 13 | 0.07 | 0.05 | 0.03 | 0.03 | 0.03 | 0.02 | 0.02 | 0.02 |
| 12 | 0.07 | 0.05 | 0.04 | 0.03 | 0.03 | 0.02 | 0.02 | 0.02 |
| 11 | 0.07 | 0.05 | 0.04 | 0.03 | 0.03 | 0.02 | 0.02 | 0.02 |
| 10 | 0.07 | 0.05 | 0.04 | 0.03 | 0.03 | 0.03 | 0.02 | 0.02 |
| 9 | 0.07 | 0.05 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 | 0.02 |
| 8 | 0.07 | 0.05 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 |
| 7 | 0.07 | 0.05 | 0.04 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 |
| 6 | 0.07 | 0.05 | 0.04 | 0.04 | 0.04 | 0.03 | 0.03 | 0.03 |
| 5 | 0.07 | 0.05 | 0.05 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 |
| 4 | 0.07 | 0.06 | 0.05 | 0.05 | 0.05 | 0.04 | 0.04 | 0.04 |
| 3 | 0.08 | 0.07 | 0.06 | 0.06 | 0.06 | 0.05 | 0.05 | 0.05 |
| 2 | 0.09 | 0.08 | 0.08 | 0.08 | 0.08 | 0.08 | 0.08 | 0.08 |
| 1 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.14 | 0.15 |

Table A.34: Balcancing Mobility Algorithm for Poisson Part 2

| | 10 h | 11 h | 12 h | 13 h | 14 h | 15 h | 16 h |
|---|---|---|---|---|---|---|---|
| 16 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.01 |
| 15 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 14 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 13 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 12 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 11 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 10 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 9 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 8 | 0.03 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |
| 7 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 |
| 6 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 |
| 5 | 0.04 | 0.04 | 0.04 | 0.04 | 0.03 | 0.04 | 0.04 |
| 4 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 |
| 3 | 0.05 | 0.05 | 0.05 | 0.05 | 0.06 | 0.05 | 0.06 |
| 2 | 0.08 | 0.08 | 0.08 | 0.08 | 0.08 | 0.08 | 0.08 |
| 1 | 0.15 | 0.15 | 0.15 | 0.15 | 0.15 | 0.15 | 0.16 |

# Appendix B

# Awards and Certificates

Awards and some certificates during the time of the PhD are attached in this chapter.

## B.1 Best paper award

At 2015 IEEE International Conference in Cloud Computing (ICCC 2015) hosted in PNU, My paper "How reduce max algorithm behaves with symptoms appearance on virtual machines in clouds", was presented in April 2015 and won the best paper in the conference.

كلية علوم الحاسب والمعلومات
جامعة الأميرة نورة بنت عبد الرحمن
Princess Nourah Bint Abdulrahman University

جامعة الأميرة نورة بنت عبد الرحمن
Princess Nourah Bint Abdulrahman University

ICCC

# Certificate of Distinction

College of Computer and Information Sciences
at Princess Nourah bint Abdulrahman University
hereby certifies that the research paper submitted by

Sultan S. Alshamrani

has been selected as a distinctive paper
International Conference on Cloud Computing ICCC15
held on 28-27 April,2015
at the University Conferences Center

Conference Organizing Committee Chair
Dr. Sahar Siraj Shabanah

# شهادة تميز

تشهد كلية علوم الحاسب والمعلومات
بجامعة الأميرة نورة بنت عبد الرحمن
أن الورقة البحثية المقدمة

من/ سلطان الشمراني

قد تم اختيارها كورقة بحثية متميزة
المؤتمر الدولي للحوسبة السحابية ICCC15
المنعقد بتاريخ ٨-٩ رجب ١٤٣٦ هـ
بمركز مؤتمرات الجامعة

رئيسة اللجنة المنظمة للمؤتمر
د. سحر سراج شبانة

المؤتمر الدولي للحوسبة السحابية
ICCC15
International Conference on Cloud Computing

IEEE

## B.2 Best presenter

I have been awarded as the best presenter in Cloud computing and security session of SSC8 conference in London.

8th SSC
مؤتمر الطلبة السعوديين في المملكة المتحدة
Saudi Students Conference - UK

# Certificate of Award
## for
# Excellence in Presentation

*Presented to*

**Sultan Alshamrani**

*at the 8th Saudi Students' Conference*

*that was held at Queen Elizabeth II Conference Centre*

*London, 31st Jan – 1st Feb 2015*

Faisal M. Almohanna Abaalkhail

*Saudi Arabian Cultural Attaché in the UK*

Imperial College London

جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology

الجمعية العلمية للطلاب السعوديين في المملكة المتحدة
SCIENTIFIC SOCIETY FOR SAUDI STUDENTS IN THE UK

وزارة التعليم العالي
MINISTRY OF HIGHER EDUCATION
الملحقية الثقافية السعودية في لندن
SAUDI ARABIAN CULTURAL BUREAU IN LONDON

## B.3 Distinguished student

I have been awarded as a distinguished student from his highness prince Mohamed Bin Nawaf the ambassador of Saudi Arabia to the UK.

بسم الله الرحمن الرحيم

سفارة المملكة العربية السعودية
لندن

# شـــكر وتـقـديـر

المكرم الطالب/ سلطان بن سعد مسفر الشمراني

**"حينما تتكاتف الهمم وتتوالى الجهود يصبح الحلم حقيقة ويصبح التميز واقعا"**

يسرني أن أقدم لكم شكري وتقديري على تفوقكم وتميزكم في دراستكم متمنياً لكم مزيداً من التقدم والنجاح والاستمرار في العطاء في حياتكم العلمية والعملية لما في ذلك من خدمة للدين الحنيف والوطن الغالي.

**وفقكم الله عزّ وجل لما يحب ويرضى**

السفير

محمد بن نواف بن عبد العزيز آل سعود

١٦ ذو الحجة ١٤٣٦ هـ

*29th September, 2015*

## B.4 Participation at conferences

I have participated in 6 conference as follows:

### B.4.1 SSC8

My paper "Discovering malicious behaviour symptoms in cloud systems" is presented in January 2015 in SSC8 and published in Proceedings of the Conference in the UK (2016), World Scientific in Imperial college press.

8th SSC
مؤتمر الطلبة السعوديين في المملكة المتحدة
Saudi Students Conference - UK

# Certificate of Participation

*Awarded to*

## Sultan Alshamrani

*acknowledging the valuable participation in*

*the 8th Saudi Students' Conference*

*that was held at Queen Elizabeth II Conference Centre*

*London, 31st Jan – 1st Feb 2015*

Faisal M. Almohanna Abaalkhail

*Saudi Arabian Cultural Attaché in the UK*

Imperial College London

جامعة الملك عبد الله للعلوم والتقنية
King Abdullah University of Science and Technology

الجمعية العلمية للطلاب السعوديين في المملكة المتحدة
SCIENTIFIC SOCIETY FOR SAUDI STUDENTS IN THE UK

وزارة التعليم العالي
MINISTRY OF HIGHER EDUCATION
الملحقية الثقافية السعودية في لندن
SAUDI ARABIAN CULTURAL BUREAU IN LONDON

### B.4.2 ICCC15

My paper "How reduce max algorithm behaves with symptoms appearance on virtual machines in clouds", was presented in April 2015 in ICCC 2015.

كلية علوم الحاسب والمعلومات
جامعة الأميرة نورة بنت عبد الرحمن
Princess Nourah Bint Abdulrahman University

جامعة الأميرة نورة بنت عبد الرحمن
Princess Nourah Bint Abdulrahman University

# Certificate of Participation

College of Computer and Information Sciences
at Princess Nourah bint Abdulrahman University
thanks

Dr. Sultan Alshamrani

for submitting a research paper

International Conference on Cloud Computing ICCC15

held on 28-27 April,2015

at the University Conferences Center

Conference Organizing Committee Chair
Dr. Sahar Siraj Shabanah

# شهادة مشاركة

تتقدم كلية علوم الحاسب والمعلومات
بجامعة الأميرة نورة بنت عبد الرحمن
بالشكر

د. سلطان الشمراني
لتقديم ورقة بحثية

المؤتمر الدولي للحوسبة السحابية ICCC15
المنعقد بتاريخ ٨-٩ رجب ١٤٣٦ هـ
بمركز مؤتمرات الجامعة

رئيسة اللجنة المنظمة للمؤتمر
د. سحر سراج شبانة

IEEE

### B.4.3 Dasc 2015

My paper "Efficient Discovery of Malicious Symptoms in Clouds via Monitoring Virtual Machines", was presented in October 2015 at Dasc 2015.

The 15th IEEE International Conference on Computer and Information Technology (CIT-2015)
The 14th IEEE International Conference on Ubiquitous Computing and Communications (IUCC-2015)
The 13th IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC-2015)
The 13th IEEE International Conference on Pervasive Intelligence and Computing (PICOM-2015)
Liverpool, UK, 26-28 October 2015

# CERTIFICATE OF ATTENDANCE FOR IEEE CIT-2015/IUCC-2015/DASC-2015/PICOM-2015 CONFERENCES

**This is to certify that *Sultan Alshamrani* has attended the IEEE CIT-2015/IUCC-2015/DASC-2015/PICOM-2015 conferences and presented the paper**

**Paper ID:** *DASC-111*

**Paper Title:** *Efficient Discovery of Malicious Symptoms in Clouds via Monitoring Virtual Machines*

**Date and place:** *26–28 October 2015. Liverpool , UK*

**Dr. Jia Hu**

**CIT-2015/IUCC-2015/DASC-2015/PICOM-2015**
**General Chair**
**College of Engineering, Mathematics and Physical Sciences**
**University of Exeter**
**Exeter**
**EX4 4QF, UK**
**Tel: +44 1392 723628**
**Web: http://cse.stfx.ca/~cit2015/**

### B.4.4 SSC9

I have participated in SSC9 by a poster in February 2016.

# Certificate of Participation

Awarded to

## Sultan Saad M Alshamrani

acknowledging the valuable participation in the 9th Saudi Students' Conference

The ICC, Birmingham, 13 - 14 Feb 2016

**Faisal M. Almohanna Abaalkhail**

*Saudi Arabian Cultural Attaché in the UK*

9th ssc

Organisers

وزارة التعليم
Ministry of Education
الملحقية الثقافية السعودية في لندن
Saudi Arabian Cultural Bureau in London
SCIENTIFIC SOCIETY FOR SAUDI STUDENTS IN THE UK

Sponsor University

Host University

UNIVERSITY OF BIRMINGHAM

Sponsors

KAPSARC
King Abdullah Petroleum Studies and Research Center
الرياض

Publisher

Springer


### B.4.5 CLOUDTECH 2016

My paper "The Impact of Hierarchical Structure on Efficiency of Cloud Monitoring" was presented in May 2016 at CLOUDTECH 2016.

### B.4.6 ECCWS-16

I have participated in ECCWS-16 by two papers in July 2016. One of the them is "Balancing Mobility Algorithm for Monitoring Virtual Machines in Clouds" as the first author and the other one is "Failure or Denial of Service? A Rethink of the Cloud Recovery Model" as a co-author.


Universität der Bundeswehr München
acpi
Universität der Bundeswehr München
Research Center Cyber Defence
CODE
Research Center Cyber Defence
CODE
ECCWS 2016
Universität der Bundeswehr München
Germany
7-8 July 2016
This is to certify that
Sultan Saad Alshamrani
Participated in the 15th European Conference on Cyber Warfare and Security
and presented the Paper:
Balancing Mobility Algorithm for Monitoring Virtual Machines in Clouds
Conference participation Confirmed
acpi
Not valid until official stamp is affixed
Reg no. 6233410
Sue Nugus, Conference Director
Academic Conferences & Publishing
International
Robert Koch
Programme Chair